A real time vector scanning multiprocessing system for image learning, identification, analysis and control

Mayes, Keith Edward

Award date:
1987

Awarding institution:
University of Bath

TITLE

A Real Time Vector Scanning Multiprocessing System
for Image Learning, Identification, Analysis and Control.

submitted by Keith Edward Mayes BSc AMIEE

for the degree of PhD

of the University of BATH

1987

Signed Keith Mayes

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

SUMMARY

This thesis describes the design, development and use of a real time, colour video, image processing system suitable for tracking and control. The system incorporates a parallel architecture of RISC processors (TMS32010s), a high performance video capture and display unit and a sophisticated vector scanning data transfer tool. Emphasis is placed on the need for "video tuned" hardware and the considerable benefits from vector scanning.

Chapters 1 to 4 detail hardware and system software design considerations. Of special note is the definition of the Vector Orthogonal Scanner* which has the rare if not unique capability of scanning images along complex loci at very high speed. The use of this unit is also discussed in conjunction with Data Transfer Processors with regards to video frame rate low level processes.

Chapters 5 & 6 consider image segmentation from both region and edge based approaches and show how this system can perform spatially variant colour thresholding and oriented edge detection and coding.

* At the time of writing the Vector Orthogonal Scanner is undergoing a patent search with a view to an eventual application.

Chapter 7 reviews existing fast shape measurements and develops a high performance shape descriptor for the vector scanning machine.

Chapter 8 investigates maximum possible rates of track for single and multiple targets.

Chapters 9 & 10 consider the diverse applications for which the system is suitable and go on to suggest the future evolution of the system based on enhanced operation and production criteria.

This work also includes copies of two published papers by the author that have been presented at international conferences.

CONTENTS

This work is dedicated with love and gratitude

to my mother and my wife.

INTRODUCTION

As a concept the real time analysis of television images is not astoundingly new, but as a physical reality often remains elusive. To expand on this statement it is necessary to consider just what is meant in this context by real time image analysis. A common description is that of a computer vision system which processes picture data within fixed time constraints. Alternatively the laymans explanation might be to make a computer understand what it "sees" in order to make a relevant decision or take some action within an allowed time limit. Neither account is particularly precise but both stress the need for a minimum speed of operation which is ultimately dictated by the practical application. Without this performance qualification the use of the "real time" description is ambiguous making no distinction between machines that produce results every millisecond and those that take ten years!

The ambitious goal for this research was to design, produce and use a machine for digital image analysis, capable of processing television pictures at a rate of 25 Hz, which is the standard video frame update frequency. Such a high speed is exceptional and normally only achieved by the use of very dedicated hardware for well defined applications. However the proposed system was intended as a research tool with which to investigate various algorithms and therefore needed to be sufficiently flexible for fairly general use. The twin objectives of a high performance yet

adaptable image processor demanded a challenging program of work which ranged from state of the art digital electronics to the vagueries of vision itself.

The Human Eye Analogy

It is initially difficult for a human to appreciate the problems of machine sight mainly due to the endowment of an excellent binocular visual system which provides the most telling benchmark for many artificial image processes. It is therefore not surprising that early researchers (reviewed in ref. 1,2,3) strove to disclose the mysteries of the eye as a progression towards machine vision. To that end innumerable experiments were attempted both electrophysical and psychological but although much new knowledge was gained, all too few recommendations were made as to the practical design of the artificial vision system. In the long term this lead to a divergence of interest between the neurologists and those researchers that were using empirical methods to achieve practical vision goals. Nevertheless there is often considerable similarity, albeit unintential between artificial sight techniques and those of the eye. It is therefore of interest if not essential to compare the functions of machine algorithms with those of animal visual systems irrespective of the detailed practical implementation. Two illustrative examples relate to object detection for which it is claimed that the eye can detect edges and lines as retinal primitives and furthermore there exist Ganglion

cells (ref. 4), which respond to specific moving objects. The machine parallels are not hard to find with edge operators and line detectors abounding in image segmentation techniques. Whilst target identification and tracking systems (artificial Ganglion cells) are less common they have existed in various forms for a number of years. The electrical analogy to the retina itself is the memory array containing stored spatial measurements of scene parameters acquired from some transducer. This is an area where artificial systems have advantages over the eye in that they are not restricted to the visible spectrum and can use active as well as passive sensors.

Types of Sensor

Any wavelength of the electromagnetic spectrum shown in Fig (1) could in theory be used in a particular sensing device, but in practise severe attenuation during transmission through the atmosphere limits or prevents the use of some regions. Transducers operating in the NIR or visual regions such as conventional and image intensified T.V. cameras are simple and inexpensive and were selected for this research effort. More information about alternative sensors is given in Appendix (A).

System Throughput Rates

As previously mentioned real time operation always relates to application and in terms of image analysis requires that decisions are made about incoming picture arrays within a time allowance but

just as importantly, the process must be repeated at the specified rate. This is dictated by two limits, maximum and minimum speed. The first is how often pictures may flow through the system which if processing limitations are ignored is ultimately dictated by the sensor characteristics. The second criteria is the worst case speed of repetition at which the application control can just be maintained. Attempting to make the two limits equal for sensor rates in the 10-60 Hz range calls for powerful high speed computation.

Computation Requirements

To appreciate the immense processing demands it is worthwhile considering a representative example. A 512 by 512 image of 8 bits depth (which is a common configuration) requires some process that calls for a modest ten machine code instructions per pixel in a time of 40 mS (PAL system frame update rate).

Therefore

Instructions per Pixel	=	10	
Number of Pixels	=	$512 \times 512 = 2.5 \times 10^5$	(1)
Total Number of Instructions	=	$10 \times 2.5 \times 10^5 = 2.5 \times 10^6$	(2)
Instructions per second	=	$2.5 \times 10^6 / 40 \times 10^{-3} = 6 \times 10^7$	(3)

Therefore a trivial task requires 60 MIPS which is many times the capability of general purpose processors. In practise the number of pixels of interest can be reduced where permissible, but to counteract this the number of instructions per pixel is usually higher, sometimes very considerably so.

In defining a computer for real time (40 mS cycle) image analysis it would seem from speed demands that some form of parallelism would be essential. However, the choice of individual processing element (PE) is by no means obvious and apparently subtle differences between devices may assume immense importance when embedded in a final system. Infact there are very few formalised criteria for PE suitability, the most notable being the provision of the binary logical operators and fast cycle times.
In addition factors such as size, cost, development support and availability must be considered. The right balance of flexibility is also important as the best processing unit for a specific task is unlikely to provide a very general purpose machine and vice versa.

Considering the substantial difficulties involved in producing an image processing system, it was thought prudent to review available computer systems before embarking on an original design. The first question asked was, where was the image computer or processing element to be found? In the realm of supercomputers there can be

found some very high powered beasts indeed. If the CRAY 1 (ref. 5) is assumed representative of this group then its 80 MFLOPS capability would seem very attractive. However this supercomputer also has a super price tag and requires a refrigeration plant to keep it cool. These last two features would discourage all but the most masochistic of millionaires from attempting to build the CRAY 1 into a prototyping system.

More generally available computers such as VAX and PDP11 are primarily designed as work horses for off-line analysis and are therefore not geared towards very high speed real time operation.

A radical departure from the Von Neuman architecture is the Single Instruction Multiple Datapath (SIMD) array which promises to be an important building block in the long term future of image processing. Various types exist or have been proposed which demand a brief review if only for the interest they create in the pattern recognition community.

SIMD Arrays

The idea of making a SIMD array from many simple processing circuits was proposed back in 1958 by Unger (ref. 6), however the practical implementation of such systems is more recent taking advantage of the increased sophistication of VLSI technology. Several attempts have been made to realise these bit serial

processors, although there is much common ground between designs generally comprising a simple ALU ram, registers and multiplexers. Each unit forms part of a large interconnected grid with direct relationships between pixel areas and processor grid location as shown in Fig (2). Some of the more well known attempts are listed below:

1. CLIP4 (ref. 7, 8, 9, 10, 11). This was the earliest practical array processor chip and was therefore subject to technology limitations of the time, however the design complexity of the PE is comparable with later attempts. Quite a number of devices have been produced and used. The device was subsequently modified to take advantage of improved fabrication facilities and to improve data flow and became the CLIP5.

2. MPP (ref. 9, 11, 12) With the grand title of Massively Parallel Processor and the combined might of NASA and the Goodyear Company, a large and ambitious design might be expected and quite rightly so. The MPP is a 128*128 grid of 16,384 PEs with single bit data paths. By combining sub images it is intended to analyse images as large as 6000*6000.

3. NTT (ref. 11) Array Processor, not to be outdone by the American effort, the Japanese have come up with an IC containing an 8*8 pe array and 6K bits of memory.

4. DAP (ref. 9, 11, 13) Designed primarily as an add on to the ICL2900 mainframe few details released.

5. PCLIP (ref. 11). Proposed 8x8 grid, although each PE deals with data, from 8 pixels.

6. GRID (ref. 11) This would seem to be an attempt by GEC to rationalise the development of future PEs, by providing a library of standard design blocks, although the configuration has many similarities to other designs, it is rich in interconnection modes.

7. LIPP (ref. 11) Perhaps the most novel feature of this device is that in addition to the ALU and particular data multiplexer arrangement there are variable length shift registers and a look up register. It is boldly predicted that for a test favouring the MPP the Lipp would be a factor of ten more powerful. This interesting claim cannot be practically substantiated, as at the time of writing the LIPP exists on paper only.

To generalise, the SIMD arrays provide a means of simultaneously operating on the data of a whole picture, albeit with the same instruction. They are ideally suited to pixel level processing and in particular neighbourhood operations where a pixel is considered with respect to its nearest neighbours.

The ability to perform such primitive functions efficiently is of great use in early processing when areas and edges need to be enhanced, however in the later stages of identification, tracking and control a higher level of processing is required than a SIMD machine could provide alone.

It must be emphasised that a SIMD array is not a complete image processor in itself but requires the support of other processors for function control and to provide higher level decisions and analysis. It is also disturbingly unclear how the arrays could maintain the continuous video flow so vital for realtime (40 mS) systems. If arrays are used which are smaller than the image then some memory transfer tool is required to move the array around the image or expand and contract it using reduced resolution techniques. The SIMD array then, although an elegant and fast low level processor, cannot work without special hardware and other computers and there remains a question mark over its high speed I/O capability.

However the biggest disappointment of SIMD devices is that they are almost impossible to obtain for research purposes, even when they are physically in existence. The problem is that the majority are subject to military and commercial resitrictions. The CLIP4 is slightly more accessible but unfortunately is only available within a complete off-line image processing system (ref. 10), rather than as individual processing element chip sets.

There are alternatives to the SIMD and Host Computer approach, many of which fall into the MIMD (multiple instruction multiple data path) category. These are often multi microprocessor systems and are more able to cope with the higher level processes, although historically they are characterised by data flow problems especially when operating off a single bus system. If this constriction is eased by the use of a special purpose "video tuned" hardware environment then the MIMD machines offer a number of advantages in practical systems, such as lower initial cost, expandability, fault tolerant rerouting, and off the shelf technology. The advantage of using multiple microprocessors for low to medium level picture processing strategies is that several different tests may be running simultaneously because fortunately image analysis is inherently suited to parallelism as it can be readily broken into subtasks.

Given the vast ranges of computers and image processing, specifications were needed to define the practical system implementation more precisely. In particular the goals of the machine and its working constraints. The actual requirements called for a colour visual image processing research tool capable of performing real time techniques such as target tracking yet supporting a broad range of picture processes. Furthermore the machine was required to be relatively inexpensive, use off the shelf technology as well as allowing the facility to incorporate

new devices or hardware in future development. Any thought that the system could be centred around mainframe or minicomputers was immediately dashed by the stipulation that it should be sufficiently compact so as not to preclude the possibility of future mounting on a small vehicle.

The decision to stand up to such a daunting arsenal of constrictions and ambitions was inspired by the avidity of both academic and industrial sponsors. The desire for a physical working system was prompted partly by the wishes of Bath University and Honeywell Leafield Limited to gain more knowledge about such systems and their techniques, but also to fill a gap between the very expensive high speed military equipment and the low speed low resolution commercial units. The void exists, probably because historically, technology has always lagged behind the ideals of many researchers who have then drifted into analysis off-line. While this is a valuable means to an end there is a tendency to remain prognosticating about future systems rather than taking the higher risk option of producing a current system with existing technology. Consequently many machines for which a great number of applications exist, never make the transition from the drawing board to reality. Another danger of off-line analysis is that with the abstraction from actual binding time constraints, techniques which if attractive in terms of mathematical elegance tend to be far removed from efficient practical implementation.

Having resolved to create a real time (40mS) system with the available technology the associated restrictions had to be appreciated. Investigated algorithms could not be overly complex and in any case minimise the number of operations by pixels product. Even so, such high rate processing could only be possible if the software algorithms were very closely coupled to the hardware architecture and the flow of visual data. An early algorithmic casualty of the approach was the Fourier Transform (and its inverse). This is a mathematical means of dealing with the duality of the time/spatial and frequency domains. Classically functions such as filters are specified as frequency responses.Therefore by this route, to filter a time varying signal requires first that it be Fourier transformed, then multiplied by the transfer function and finally the output inverse Fourier transformed. If filtering is performed via this procedure the majority of the time is spent transforming this way and that with the filter being a trivial multiply. For a 1-D signal the number of primary operations required for an FFT is approximately given by,

$$OPS = N \times LOG_2(N) \qquad (4)$$

where

N = Number of samples in sequence

OPS = Number of Operations

For a (2-D) signal the equation is

$$OPS = (MxNxLOG_2(N)) + (NxMxLOG(M)) \quad (5)$$

where

N = Number of Pixels per Line

M = Number of Lines

OPS = Number of Operations

If it is assumed that N = m = 512 then the total number of operations is nearly 5 million. Bearing in mind that these will likely involve floating point maths on complex values, even with special hardware a large number of instructions will be required per operation. If 25 is taken as a very conservative estimate, then around 125 million instructions would be required. As both the transform and its inverse must be performed this value must double up to 250 million. To achieve this at the target rate of operation will require (ignoring the actual function) that these operations are completed within 40 mS.

i.e.

$$\text{OPS/Per second} = 250x10^6/40x10^{-3} = 6.25x10^9 \quad (6)$$

This means that the hardware must be capable of roughly six thousand million instructions per second! This most definitely puts the 2-D FFT way beyond the reach of the target system. In

fact even the most state of the art FFT boards (ref. 14) take several seconds to operate. When the problems of the 2D FFT were first recognised in this research the question was asked, was the transform a necessity for an image procesor? Fortunately the answer was an emphatic "no". Equation (7) shows the relationship between convolution in the time domain and multiplications in the frequency domain.

$$G(W) \times H(W) = g(t) * h(t) \qquad (7)$$

As it is mathematically difficult to deal with convolution usually the left hand side of the equation is utilised. However digital hardware can be very adept at performing convolution and in fact processors exist (ref. 15, 16, 17, 18, 19) which are specifically designed to excel at this function. Therefore image filtering can be readily achieved at high speed by convolution making the FFT approach redundant.

Having decried the poor old Fourier Transform I hasten to add that its use is invaluable for general signal processing techniques especially spectral analysis, provided of course that time is not of the essence. A simple method of obtaining a 2-D Fourier transform is given in Appendix (B) for the benefit of readers with off-line investigations in mind.

Autonomy

The eventual aim for a real time high rate image analysis system is to provide some autonomous action and/or control to replace some manual operator function.

To appreciate the motivations for this and the associated problems, it is representative to consider the work on vehicle systems. Researchers in this field soon became aware of the giant leap from operator control to fully autonomous operation and therefore proposed a program of rational development through many stages of partial autonomy. An example of this is the work done at RARDE (ref. 20) where four major stages of vehicle evolution were identified:

1) Manned
2) Remote Control
3) Supervisory
4) Autonomous

A disadvantage of manned military vehicles is that they are designed by necessity as a compromise between two goals, keeping the operator alive and completing the mission. By contrast un-manned vehicles are dedicated solely to completing their designated task and as a consequence can be less armoured, smaller, cheaper and able to function in hostile environments. An example of this is RARDE's "wheel barrow" which is basically a remote

controlled robot with a mounted camera used with much success for bomb disposal. Obviously remote control systems require continuous communications between the operator and vehicle, which from a military standpoint is a vulnerable weakness in wartime situations. Therefore a trend towards autonomous action would seem desirable. Applications within this field currently attacting attention include:

a. Sentry vehicle
b. Follow my leader convoys
c. Decoys
d. Laser Designators
e. Reconnaisance
f. Surveillance
g. Mine clearing/laying

These are all examples of supervised or limited goal autonomous machines.

A totally autonomous robot capable of adapting to and participating in any battle situation throughout the duration of a war, thankfully only exists in science fiction, at least for the present!

The limitations on full autonomy are not merely technological. If manual input is totally removed, the software strategies for decision making become extremely complex requiring artificially

intelligent systems. This is an area where a great deal of work is required for which funding is only likely to arise from high priority military, or very lucrative commercial applications.

The implications for the research tool was that only limited goal autonomy was feasible and moreover the inclusion of a man machine interface for supervisory or semi-autonomous action would make a great many applications practicable and hence enhance system capability.

Defining a Final System

Selecting the various hardware sub subsystems of the research tool was always a difficult compromise, none more so than the choice of computing unit. The fastest solution to a given problem will always be dedicated hardware implementation but such an arrangement would not be sufficiently flexible for the usage intended. A short list of four candidates was drawn up including SIMD arrays, multiple general purpose processors, bit-slice and multiple digital signal processors (DSPs). As mentioned previously SIMD arrays are limited to very low level processing, however one would certainly have been at least considered for inclusion within the system had it been available. The parallel use of general purpose microprocessors on a common bus is a widespread and relatively inexpensive means of providing increased computing power. Unfortunately the serious bus bottlenecks coupled with the relatively low performance of general purpose microprocessors,

makes the arrangement unsuitable for high speed image processing. Bit-slice designs can certainly work much quicker, but they are complicated to design and with microprogram word lengths of 128 bits wide not uncommon, they are by no means easy to program.

The most promising configuration made use of parallel DSPs and in particular the Texas Instruments TMS32010 (ref. 15). This high speed reduced instruction set device with its ready availability and development support, was finally adopted as the system PE. Fundamental to this choice was the provision of powerful instructions to implement convolution, correlation and general filtering. However the device in common with much conventional signal processing hardware, is optimised for accessing sequential data streams.

To harness the full power of such devices necessitated a means of dealing with 2-D spatial images as a number of 1-D wavetrains, similar to those produced by a raster scan (shown in FIG. (3)). If data is transferred from a predefined image area to a window memory associated with a DSP using a horizontal line scanning techniques then that window's contents are optimised for tests which search for horizontal trends. Similarly by scanning parallel to a vector allows trends along that direction to be analysed. The geometric significance of this is that the image is apparantly rotated with respect to a particular test without introducing software overheads and whilst maintaining the sequential data stream. Therefore the provision of a frame store high speed vector scanning unit was

recognised as a vital requirement for the system, without which the processors and the data streams would be mismatched. The fact that the powerful and flexible Vector Orthogonal Scanner (VOS) was physically realised despite immense practical design difficulties is indicative of the importance attached to it's functions. Also the data environment thus provided for the TMS32010 is also suitable for dedicated hardware and alternative DSPs thereby supporting future evolutionary development.

By embedding the multiprocessor computer and the VOS within a "video tuned" hardware framework VOSTTAC1 (ref. 21, 22) (APPENDIX C & D) was born. The prototype machine was capable of Vector Orthogonal Scanning, Target Tracking, various forms of Adaption and could produce both digital and analog Control signals. The block diagram of Fig (4) shows the main constituents of VOSTTAC1, a PAL source (although NTSC tolerated), high speed digital video capture and display system, parallel signal processing unit and the Vector Orthogonal Scan data transfer tool.

Of paramount importance during the design phase of VOSTTAC1 was maintaining the flow of visual information through the system, whilst not adversely affecting processing speed or flexibility. This was always a compromise between encouraging identified peak processes and easing restrictive bottlenecks.

The hardware no matter how highly specified would have been ineffectual without an operating system executive and software utilities to not only drive the system but also cope with parallelism and temporal events of varying importance. The special hardware and its associated system software are discussed in much greater detail in Chapters 1 - 4.

Having designed and produced an image processing machine the obvious course of action was to use it. In particular various established processes were evaluated in order to identify strategies most harmonious with the equipment. Chapters 5 - 8 journal these investigations, including target tracking and recount some techniques peculiar to VOSTTAC1. The abilities of the system well founded, Chapter 9 goes on to review the many application areas for which it is suitable. With an eye towards commercial applications Chapter 10 is concerned with the possibility of ruggedising and cost engineering the hardware and on the basis of experience gained proposes successors to VOSTTAC1.

CHAPTER 1

THE DIGITAL VIDEO SYSTEM

1.1. TV FUNDAMENTALS

If a colour digital video system is to be used for research purposes, it is important that it be able to accept video signals from a variety of different sources. Fortunately the need to standardize television equipment was recognised long ago and therefore new hardware is only concerned with decoding from or encoding to a required standard. Predictably as with most other matters, the countries of the world could not reach universal agreement on the colour television broadcast system to adopt, which resulted in basically three different types, NTSC, PAL and SECAM. Before considering the differences between the systems, it is necessary to first consider the way in which a video picture is acquired and then redisplayed.

Conventional television camera tubes contain a light sensitive surface, the electric charge distribution of which varies with the reflected light from the observed scene. By scanning an electron beam over this layer, the charges can be converted to a voltage change across the beam current resistor, which is effectively the video signal. The beam is deflected in a "ZIG-ZAG" fashion called a raster scan, a simple example of which is shown in FIG.(3). It may be considered as a fast horizontal line scan and a slow

vertical field scan. The beam is inhibited or "blanked" during flyback. At the display end, a cathode ray tube works in a similar fashion except that the scanning beam is used to generate light by exciting phosphors. Obviously to accurately represent a camera observed image on a CRT or video monitor requires synchronism between the rasters.

To achieve this, line and field synchronizing pulses are incorporated within the signal which is then often termed "composite video". A complication of the raster process is the use of interlaced scans. This requires that a complete picture be made up of two fields one odd and one even, FIG. (5), which occur consecutively. An inherent half line difference between the two, causes the fields to interlace. The reason for this is to save bandwidth without introducing a subjective flicker effect, (ref 23).

Typical waveforms for the PAL system (which is the British standard) are shown in FIGS. 6 and 7, in which signal amplitudes are proportional to picture luminance or grey scale information. Up to this point all systems are fairly similar, but the encoding of colour information is where they differ. The National Television Systems Committee of the USA (NTSC) basically phase modulates colour related signals onto the waveform to be eventually compared with the colour subcarrier burst reference also included within the signal. The problem with this approach is that any phase difference during transmission can result in hue variation

and hence the NTSC nick name "Never The Same Colour". The British system reduces this effect by using the "Phase Alternating Line". This method reverses the phase of the colour subcarrier on alternate lines which effectively averages out the absolute phase error. The Secam system (Sequential Colour A Memoire) which is the norm in France and Russia is different again. This system generates two colour signals which are transmitted sequentially, by contrast with PAL and NTSC which send both colour signals simultaneously. Because of this a memory component is required to allow the combination of the current and previous line hue information.

The PAL system was the obvious choice for the research rig, being the British standard and was therefore adopted as such. However, with a view to possible future commercial exploitation it was thought unwise to preclude the American system. As a consequence it was decided to operate the rig from R G B inputs and timing with decoders being considered as front end peripherals. Comparison of PAL, and NTSC shown in table (1) reveals much common ground between the actual timing, especially the line frequency. Although NTSC is generally lower resolution than PAL, its field rate of 60Hz allows higher luminance levels to be used before perceived flicker, than that possible with the slower PAL rate of 50Hz.

1.2 VIDEO SAMPLING

Having decided on the class of suitable video sources the next stage was to consider sampling, as a prerequisite for analysis by digital computer. The normal approach which was not departed from

samples the active line information as it arrives via the line scan (ref. 24, 25, 26). The spatial significance of this is equivalent to a rectangular array of delta functions being convolved with the image. The dimensions of this array are fixed by the spatial sampling rates, which are user defined for a given performance. Table (1) shows that the PAL system has approximately 577 active lines which is therefore a convenient vertical sampling rate. The horizontal rate is not so clear cut and if the IBA (ref. 27) standard for digital systems were to be rigidly adhered to a horizontal sample rate of 14.3 MHz would be required. Such a rate could be used with broadcast quality television equipment with bandwidths up to 7 MHz. However, low cost video cameras and tape recorders are unlikely to exceed $2\frac{1}{2}$ or perhaps 3 MHz bandwidth and therefore such a high sample rate would be unwarranted. The required rate for VOSTTAC1 was calculated to give the equivalent number of samples on a line as there were lines within a field subject to aspect ratio, i.e.

$$\text{SAMPLES/LINE} = (577/2) \times (4/3) = 384 \qquad (8)$$

(577 * 384 was also the specification for a Hitachi colour solid state imager which the system was originally designed to use.)

This number can be used to calculate the sample rate

$$\text{SAMPLE RATE} = 384/(a_2 \times L) = 7.4 \text{ MHz} \qquad (9)$$

where a_2 = ACTIVE LINE FACTOR = 0.812

L = LINE PERIOD = 64 uS

Low spatial resolution due to inadequate sample rates causes an effect known as posterization of which varying degrees are shown in PICS 1 to 4.

It is important to remember that if high quality sources are used, that they are bandlimited otherwise an effect called aliasing will occur due to violation of the sampling theorem (i.e. sample at least twice the frequency of the highest frequency component). A special form of this is called a De Moivre pattern, which is apparent in regular structure.

The number of bits used to store a value at each pixel location is a very important feature. Too few levels causes quantization effects to become more noticeable, too many is wasteful and expensive. Examples for 1 to 4-BIT coding are shown in PICS 5 to 7. Storage was actually provided to a depth of 12-bits configured nominally as 4 bits per primary colour e.g. R,G, & B. The signals themselves are in fact sampled and latched to 7-bits each, which may one day be required for higher accuracy.

Before any sampling can occur the PAL signal must be decoded and split into RGB and timing components. Practically this necessitates the use of a PAL decoder, but fortunately one was included within the display monitor. By careful modification it was possible to tap off timing and R G B signals and later reintroduce the modified RGB from the rig. This is shown in the block diagram of the digital video system, FIG. 8. Timing signals in the form of field synchs and composite blanking are bussed to the main system board PIC 8 from which a synched clock is derived and fed to the ADCS. The sampled RGB signals are then routed through the system, stored within the framestores PIC 9, before eventual recombination and display via the DACS on the video digitizer amplifier board PIC 10 and monitor output.

1.3. VIDEO SIGNAL LINEARITY

The linearity of the video buffering sampling and reconstruction is of interest especially with respect to the relative tracking of the three primary colours. Poor colour tracking would result in false hues being produced which may for example make a greyscale image appear coloured. To counteract this the analog video buffering circuits have a number of gain and shift adjustments to ensure equally calibrated channels. However, camera tube sources are inherently non-linear a measure of which is called GAMMA. This effect will differ from one camera to the next as will the in built GAMMA correction circuitry. Further complications arise from the use of different sensitivities of tube for the primary colours ranging from most sensitive green to least sensitive blue. The

reason for the differences is primarily economic and the ordering arises from the relative effects of a primary colour on the luminance signal, i.e.

$$Y = 0.3R + 0.59G + 0.11B \qquad (10)$$

Where, Y = LUMINANCE SIGNAL

Another source of non-linearity can be the type of illumination. For an increase in white light illumination such as sunlight, there will be a corresponding proportional increase in RGB. However if the lighting is artificial or filtered it may not contain the full visual spectrum and therefore an increase in illumination would not have the expected effect. In practical terms the major effect of the errors and non-linearities mentioned is to prevent a particular colour combination from accurately tracking over the whole luminance range which is especially noticeable at low light levels. In fact low lighting also introduces optical degradations. Most cameras are equipped with automatic irises, which when faced with very dim scenes, compensate by fully opening the aperture. Unfortunately, due to diffraction effects there is an associated shading around the perimeter of the image which can be quite severe. This can be counteracted by the use of "U" shaped horizontal and vertical post-emphasis amplifiers if required. The actual front end conditioning is application and to some extent source dependent, usually requiring considerable effort to create a linear colour system.

1.4. FRONT END CONDITIONING

Provision has been made for the eventual inclusion of front end conditioning functions, possibly in the form of large programmable array logic devices or Pals (not to be confused with the television system). There are 3 slots provided for through line modification of the 3 MSBS of each colour and another 3 for observing incoming data. Possible functions which may be achieved with Pals or analog circuitry might be thresholding,filtering, contrast enhancement or perhaps mean removal. Early thresholding is probably the most common but it has two main disadvantages

1) Its effect is global, i.e. every pixels within a store is affected.

2) The actual picture is not stored so information is lost.

The technique is popular because it can threshold a vast amount of data yet only introduces a small time delay. A software approach by comparison would require at least one instruction per pixel and with image sizes of 512 x 512 not uncommon, a minimum of a quarter of a million instructions would need to be dedicated to this function. In VOSTTAC1 thresholding is performed in hardware but at a later stage and in such a way that the stored picture is not corrupted and spatially variant thresholding supported.

Possible enhancement filters might be those for aperture shading correction or perhaps log transfers to counteract Hurter and Driffield emulsion curves, FIG. (9) and thereby enhance photographic contrast. Local line mean removal is also a useful process when dealing with I.R. sensors, however the constraint of computing the mean along one particular line direction may be too much of a restriction to allow general use.

A display effect which is not directly related to input data can be produced via the auxilliary video display unit PIC (11). This consists of a low resolution 1-bit video mapped memory controlled by a Z80 microprocessor (ref. 28). Its effect is noticed by producing an EXOR function with the MSBS of the main video prior to display. It is useful for displaying tracking crosses, messages and when used in conjunction with a light pen input becomes a man machine interface suitable for target nomination.

1.5. TIMING SYNCHRONISM

Timing synchronism has been mentioned briefly as a requirement for signal sampling, but furthermore it is vitally important for complete system operation. In a television studio there is no great problem, one signal generator is used to provide timing inputs for all the equipment. However cheap VTRS, cameras and computers are unlikely to have a timing input and therefore synchronism of any attached digital video system must be derived from the composite video waveform. Two types of synch pulse are of interest, line (horizontal) and field (vertical). The first is of

interest when generating the sample clocks, to digitize only the active parts of the line, shown in FIG (6). The second is important if picture information is to be stored in a consistent way in memory. Field synchs occur during field blanking and can be used to indicate the beginning or end of a field, but do not convey any information as to which is the current field. Analog television does not need to know if the current field is odd or even providing there is a half line difference between the last field to ensure interlace. This difference of a half line can be tested to identify fields and hence allow precise digital storage. Ignoring field differences for the moment, sampling would proceed as follows. On start of active line the sample clock and sample clock by two are both enabled and commence with rising edges. A count of samples per line is maintained until it matches a number stored within an end of line prom (from which several different line lengths may be selected). When a match occurs the clocks are inhibited but in such a way as to maintain complete cycles.This last point is very important as signals are derived from the sample clocks which eventually generate valid memory address strobes for the frame stores.

The mechanism used for line sampling or its violation can be used for field lock in. If for example a full line is expected, but the half line present at the start of the even field is present, then no end of line will be detected before the next line blanking

pulse. This can be flagged as an error event which indicates that the fields are logically swapped and causes a changeover. The actual procedure is as follows:

1) At the start of the even field load count with a half line size value and then increment to full value, causing an end of line flag.

2) Load count with full line size value and increment to double length to generate end of line.

3) Repeat 2 until next field sync, then logically toggle between fields.

4) Load count with full line size value and increment to double length to generate end of lines.

5) Repeat 4 until next field sync, then toggle between fields and go back to 1.

If any start of field causes an end of line error flag then the fields are logically toggled. This helps to make the system more robust with respect to glitches and will in fact recover even if the video source is temporarily disconnected. This method works without modifications for all true PAL systems, but not all those

claiming to be PAL, posing the question, when is a standard not a standard? The answer is when it is cheaper or easier for a manufacturer to bend the rules.

A problem when using cheap sources is that the timing signals tend to depart slightly from the ideal, common examples are variation in active line length and the number of active lines. A more subtle problem which effects field lock is when the half line difference between fields is produced by say a quarter line in the even field and a three quarter line in the odd field rather than half and full. Fortunately this maybe overcome by producing a start of field gating pulse which limits tests to beginning of fields followed by a monostable extended blanking which can strip off the starts of fields until a half line, full line situation is restored. Another irregularity is the use of non-interlaced scans, usually found on computer type outputs, but even PAL test signal generators have been found to be culprits. Although these signals do not adversely effect synchronism due to the self-correcting field locking it is important that any processing is made aware of the redundant field. Differing numbers of active lines are automatically compensated for but with varying active line lengths it is important that the numbers of samples required can actually fit on the line, otherwise synch lock will fail. To avoid this, lines are sampled at the rate at which 384 samples should fit on a line, yet only 360 are taken allowing a reasonable safety margin.

Picture memory may be thought of as a 1-D array containing 229376 locations which can be partitioned for any line length and sample rate by varying end of line prom selects and switched values for half and full lines. Reduced sample rates may be tolerated by using lower divisions of the ADC clock to produce valid memory address strobes for which rising edge synchronism and full cycle termination is provided.

The timing and data sources for the digital video system are generally external, but a PAL compatible timing unit is incorporated on board to assist debugging or to act as a good quality synch generator for external peripherals. Alternative data paths are also provided, facilitating the use of a horizontal bar pattern, fixed colour screens as well as external data. All the sourcing modes are selectable under software control.

1.6. FRAME STORE ADDRESS GENERATION

Frame store address generation is an essential feature of the main video board and allows storage and retrieval of sampled values. If the data was packed away sequentially as it arrived the memory would contain one field after the other in address space. As a consequence there would be a large memory space to be traversed to access similar lines within alternate fields. A requirement of the VOS machine (see Chapter 3) was that like lines be adjacent in memory, which necessitates the use of interlaced storage. In terms of hardware this requires a state machine capable not only of address increments but jumps as well. The flow chart of FIG. (10)

shows the address generating process for interlaced field storage. For every access the machine generates an address, a read/write signal and a valid memory address strobe, which are directed to one of the two frame stores designated P and Q.

The reason for every frame store pixel being directly addressable had little to do with video constraints, but arose from a requirement of the VOS machine for high speed random access, necessitating static memory. Apart from their high speed the stores are unusual in the way in which they make best use of the rams. The requirements called for high speed, low power consumption and random access. Examining the characteristics of the chosen device when working off a tri-state bus revealed a number of problems. A poor output specification would allow for a recently switched off device to be active for up to 30nS after deselection, causing chip contention. In extreme cases this could cause damage, but in any case would generate noise as the output buffers struggled for surpremacy. A further problem was that the time taken to bring the ram from standby and access it using the chip select was 5nS longer than the continuously enabled time. In fact manufacturers revealed that the 45nS rams were unlikely to be used within cycle times less than 100nS. This rate was unacceptable so a "trick" was sought which would provide continuously enabled access times, minimize output contention, maintain the majority of ram in powered down state yet supported random access. At first

glance the problem appeared insurmountable, yet closer investigation of the required so called random access yielded a lever with which to crack the problem.

1.7. Pseudo Random Access.

Consider the diagram of FIG. 11 showing a vector scan being used to transfer a block of memory. All such scans procede by means of line increments (LINCS) until the designated line length is achieved when a jump to the start of a newline is performed (NLINC). For a typical 90 x 90 window there will be 8100 transfers involving 89 NLINCs. In which case approximately 1% of the address generations require NLINCs and 99% use LINCs. In practise there is often considerably fewer lines of wider width such as in full screen bar mode, i.e. 360 x 22 corresponding to only 0.3% NLINC jumps. Therefore if LINCs can be optimized then so is the overall transfer.

If the rams comprising the memory array are selected sequentially, then ignoring incomplete lines, they represent horizontal picture bars containing 16K of pixels for the devices used, with the memory organised in such a stream the jumping capabilities of the LINC operation within this address space was investigated. One of the largest LINCs is likely to occur when say a 90 x 90 window is used under reduced resolution to represent the whole image by means of line and pixel skipping, approximate values of which are shown below for a vertical scan:

PIXELSKIP FACTOR = INTEGER (LINESIZE/WINDOW RANK) = 4 (11)

WHERE LINESIZE = 360

WINDOW RANK = 90

LINESKIP FACTOR = INTEGER (NUMBER OF LINES/WINDOW RANK) = 7 (12)

WHERE NUMBER OF LINES = 577

LINC = LINESKIP X LINESIZE = 2520 (13)

NLINC = PIXELSKIP = 4 (14)

The maximum LINC in this case is well within the 16K range of a ram device which could in fact cope with a lineskip factor of 45 for a 360 line. The significance of this is that from the last scan point a LINC will cause the address either to stay within the confines of the current chip or move into the address space of the one immediately below it. In other words it is incapable of jumping completely over a chip without accessing it. At the end of a line this condition is usually violated when large jumps occur in

address space relative to the last point, i.e. NLINCs. One can therefore define a pseudo random access to be mainly a read or write within the address space of the last accessed ram or the next one within the stream but with the rare occurence of large jumps in address space. From this definition it was then possible to design hardware to satisfy earlier criteria.

To use a continuously enabled access mode requires that a ram be powered up prior to its use. Pseudo random access therefore presented a problem as the current and next chip must be powered simultaneously which could not be tolerated on a single bus. The solution was to use a double bus, the chips of which were interleaved in memory space, FIG. 12. In this way the required rams could be powered whilst maintaining the rest in low current standby mode. A scheme was then needed for maintaining the powered up or down status of the frame store memory. In practise this was achieved by using two asynchronous state machines called Power Up Predictors which used the MSBs of the address to anticipate when to activate particular ram banks, see FIG. 13. This supported full speed access during LINCs, but whenever a flyback or NLINC occurred the prediction would fail. This was accommodated by allowing an extra clock cycle wait state during end of lines, which gives sufficient time for the state machines to recover. In block transfer mode this event is so rare that almost full speed operation is achieved. Special purpose scans will be more affected as the ratio of NLINCs to LINCs increases.

1.8. Dual Store Usage

Each frame store is 12 bits deep and comprised of 3 boards each contributing 4 bits. The stores themselves are not without processing capability as they each incorporate a Data Transfer Processor (DTP) which is discussed in detail in later chapters. However, in most applications picture information must be available for analysis by the main processors. Two stores are provided, one of which is updated with incoming video, whilst the other is accessed by slave processors via the VOS machine. It is therefore necessary to periodically change over the roles of the stores but it is vital that this operation is compatible with both processing and video timing. Therefore a flexible toggle control system was devised which supports two modes of operation, manual and on-line. The first mode is used for off-line investigations wherein the operator is responsible for filling and toggling the stores. This is achieved by using the automatic toggle on-off, the store select, and the write disable switches. The last of which provides the operator with a freeze frame facility. In on-line mode the master processor is responsible for store control and toggling. The master processor is able to synchronize to video events by waiting for video board interrupts, which if enabled occur at the beginning of new frames, and cause the frame stores to toggle over. A common method of operation is to:

a) Enable interrupts and toggle.

b) Wait for interrupt (new frame) and automatic toggle.

c) Disable interrupts and toggle.

d) Process image.

e) Go to a.

There are however occasions when using the DTP, such as in background subtract mode, when the interrupt and toggle must function independently and when there is a need to know which physical store is being accessed. Therefore suitable controls and flags are accessible under software control. Another selectable feature gives a choice between read before write or write before read mode. The first is essential for background subtraction and for displaying modified stores, the second forces the display to incoming video when current store contents are meaningless or of no interest. In cases where incoming video would also tend to confuse the store outputs may be blanked for specific frames.

The toggle flag indicates which store is currently required by the video sub system and its inverse is bussed out to the VOS machine so that the two units never contend for the same store. However, if required there are ways in which the VOS unit could "steal"

accesses from the current video store. During blanking all video buffers are switched off, which means that the VOS could in theory access either store. The amount of blanking time available can be calculated as follows:

0.812 = a_2 = active line factor

64uS = L = line period

0.922 = a_1 = active field factor

625 = N = lines/picture

T_F = picture period = 40ms

T_B = Blanking time

$$T_B = T_F - Nxa_1xLxa_2 = 10\ mS \qquad (15)$$

i.e. quarter of the frame time is redundant with respect to sampling and storage and therefore "stealable". If more drastic action is required there is a control line which although currently undriven could inhibit the video board outputs for an indefinite period.

To summarize, the digital video system of VOSTTAC 1 (ref. 29) enables pictures derived from PAL (or perhaps NTSC) composite video to be stored or displayed whilst providing memory access facilities for other hardware. It represents one of the main sub units of the system, as does the memory transfer tool (VOS). These two combined are not without processing capability due to the DTP's, but it is

at a very low level and of limited variation. Therefore the main signal processing computer, which is the third sub system reached via the VOS machine, is of great importance and discussed in the following chapter:

CHAPTER 2

TMS32010 MULTIPROCESSOR SYSTEM

2.1 OVERVIEW OF THE TMS32010

The Texas Instruments TMS32010 (ref. 15,30) 16/32 bit DSP provides a practical alternative to bit slice design for high speed data processing. It can achieve a very respectable 5 MIPS and exhibits features in common with both array processors and control devices. The high speed can be attributed to efficient pipeling methods within a modified Harvard architecture. This architecture is characterised by having separate program and data areas, allowing a complete overlap of the instruction fetch and execution as shown in FIG. (14). A variant on the Harvard approach which enhances system flexibility, is the inclusion of software instructions which effect word transfers between program and data memories. This useful facility is just part of a powerful instruction set. Single instructions are equivalent to several operations on general purpose machines and are ideally suited to sequential signal processing problems. These take advantage of the special hardware, which includes a 16 x 16 bit multiplier, 0 to 15 bit barrel shifter, 32 bit accumulator, 0, 1 or 4 bit parallel shifter and autoincrement/decrement auxiliary registers.

2.2 TMS32010 PROGRAMMING

A typical TMS32010 program example is shown below in which a sequential data stream is convolved with a 3 bit mask.

Where

Y [n] = Output Value

X [n] = Input Value

H [K] = Convolution Coefficient

Therefore

$$Y[n] = X[n] \bullet H[0] + X[n-1] \bullet H[1] + X[n-2] \bullet H[2] \qquad (16)$$

or

$$Y[n] = \sum_{K=0}^{K=2} H[K] \bullet X[n-K]. \qquad (17)$$

Practical convolution basically requires that the input and delayed versions of it be multiplied by a fixed set of coefficients and the results accumulated to form the output.

Programming Assumptions

a) Data memory contains variables X0 - X2 in consecutive and increasing addresses which act as a shift register.

b) Variables HO - H2 contain the convolution coefficients.

c) No overflow can occur.

d) "SETUP" is the specified initial conditions for the shift register.

Example Program

```
*       INITIALISE DELAY LINE CONSTANTS
        LAC SETUP,O
        SACL X1
        SACL X2

*       CLEAR ACC AND P REGISTER
CONVLP  ZAC
        MPYK O

*       GET VALUE FROM INPUT
        IN XO,INPUT

*       PERFORM CONVOLUTION
        LTA X2
        MPY H2
        LTD X1
        MPY H1
        LTD XO
```

```
        MPY HO
        APAC
        SACL RESULT

*       OUTPUT RESULT VALUE
        OUT RESULT, OUTPUT

*       LOOP BACK TO START
        B   CONVLP
```

The loop cycle time is 16 clock periods which with a 200nS period gives a maximum data rate including I/O of

DATA RATE = $1/(16X200nS) = 1/(3.2x10^{-6}) = 312.5$ KHz

(18)

The main loop only contains 13 instructions including data I/O and looping control. This compactness is due to the power of the individual instructions e.g.

MPY = multiplies contents of T register with P register (200nS)
APAC = adds contents of P register to accumulator (200nS)
LTA = as APAC but also loads T register (200nS)
LTD = as LTA but also shifts contents of specified datamemory to next highest location (200nS).

A full list of instructions is given in Appendix (E) and treated in much greater detail in reference 15.

2.3. EXPANDING THE PE

It must be emphasised that the more conventional instructions are also provided including the binary logical operators, which support the broad range of data tests and manipulations necessary for image analysis. Certainly from a programming point of view the TMS32010 would seem acceptable, but in terms of I/O the minimal PE is unsuitable due to is inability to access external datamemory which is a necessity for image analysis work. Fortunately the problem may be overcome by means of a modest expansion of autoincrement and decrement counters, implemented on the I/O ports provided. The PE is then optimised for sequential access to ram, compatible with the more powerful signal processing instructions of the DSP. Sequential tests are required for various techniques, for example, digital filtering, convolution and correlation, however many pattern recognition algorithms such as clustering and neighbourhood tests (ref. 31) require random access to external memory. Therefore this mode is supported, but incurs a small time penalty relative to the sequential option, although transfer times are still comparable with modern microprocessors running at much higher clock rates. The expansion circuit remains efficient providing the address space is fairly small (i.e.$\leq$32K) which is obviously inadequate for direct access

to the large amounts of memory associated with picture storage. It is therefore vital that a fast and efficient memory transfer and management unit be available such as the VOS machine described in chapter 3.

2.4. MULTIPROCESSOR ARCHITECTURE

It has been suggested that pattern recognition systems can be treated conceptually as cones or pyramid structures reference (ref. 32, 33) as shown in FIG. (15). The most notable feature of which is the decrease in resolution associated with an ascent of the pyramid. The structure supports three general modes of operation:-

a) Data reduction (ascent of cone)

b) Iteration (fixed level of cone)

c) Projection (descent of cone as a result of a higher level strategy decision).

Within this framework modular parallel processing algorithms have been developed successfully by other research groups which would seem to recommend the general pyramid concept. A MIMD (ref. 34) implementation would require that the individual DSPS be responsible for quite large groups of pixels (as opposed to the one to one correspondence of SIMD arrays). In which case direct hardware implementation would restrict PES to fixed layers and

hence a reduced number of PEs would degrade resolution as only the upper layers of the pyramid could be filled. In practice it may be more desirable to have the option to trade speed rather than resolution against hardware, as the former not only allows lower initial cost and effort, but also strengthens the system as fault orientated rerouting could be supported. Therefore a flexible hardware adaptive architecture is required. The chosen arrangement may be considered as an expandable tree structure as shown in FIG. (16). At the top of the tree is the master processor, responsible for the ultimate control of the slave PEs and peripherals on the lower branches. Data can reach a particular node by one of two routes either from the immediate sub master or from the frame stores via the VOS unit. The latter is not penalised by tree height as it can supply data to all levels simultaneously. In doing so it fills small picture "windows" associated with each slave, from variable screen positions and with optional resolution reduction.

2.5. THE MASTER PROCESSOR

If VOSTTAC1 was an orchestra, the master processor would be the conductor providing control and synchronism to all the constituent sub sections to achieve harmonious operation. Musical analogies apart, it consists of an expanded PE, a program memory, interrupt priority encoder, port expansion and various buffers and multiplexers. Although it has in excess of 16K expanded memory

address space it differs from the slave processors in that it has no external data memory which is dedicated to itself. However rank has its privileges and the master is quite capable of "evicting" slaves and taking over their memories.

Being the system manager and strategy co-ordinator the master requires communication channels of varying importance between itself and other system units. This is effected by means of a prioritised interrupt facility, which is slightly unusual in that it uses the BIO or "soft interrupt" line. This is preferred to the normal hard interrupt as there is less likely to be stack difficulties, but more importantly time critical loops need not be disturbed.

2.6. EXPANDED PORTS

A TMS32010 has provision for 8 input and 8 output 16 bit ports, which is inadequate to communicate with all the logical units within VOSTTAC1. Therefore a 2 to 16 port expansion is used. I/O is achieved by writing a logical unit number to one port to open a stream and then directing data to and from the unit along that stream via the second port.

Physically the master is distributed over two sub-units, the main processor board, PIC 12, and a smaller motherboard, PIC 13, which contains the priority encoder, datamemory expansion, port expansion and buffers. The motherboard handles the master

expansion bus to slaves and peripherals, which includes a 16 bit databus, port address strobes and the multiple level interrupt lines. Circuit details can be found in reference 29.

2.7. SLAVE PROCESSORS

Similarities between slaves and the master processor board are not unintentional. The major differences are the lack of slave expansion boards and the inclusion of on board slave data memory. (The lack of master data memory is due to space restrictions which would not be the case if a more modern version was constructed).

The slave boards also have a peripheral connection which is directly compatible with that of the master PE prior to its motherboard. Therefore peripherals can be used at different levels or locations within the system. Current examples are a multichannel analog I/O controller board (ref 35) and a VOSTTAC1 TO BBC model B interface, PIC 14 used for picture storage and retrieval.

The master processor is very rarely "ruffled" by events external to it. Without a hardware superior its operation may only be redirected as a result of "polite" requests from slaves and hardware via the soft interrupt lines. A slave by contrast has a much more hectic time, with its vitals held by the master i.e. soft and hard interrupts and reset line. As if this wasn't

sufficient exploitations, these same control lines are controllable by any peripheral connected to a slaves I/O bus. Therefore to preserve some order an event origin status word is generated which at least identifies to the slave the current instigator of frenzied activity. Fortunately for the hardworking slave there is always the possibility of promotion to a sub master.

Currently only a two level tree implementation is used consisting of a master and 4 slaves, however it is theoretically possible to expand the tree downwards via a hierarchy of sub masters. The transition from slave to sub master requires the addition of a motherboard identical to that used by the ultimate master. Apart from the extra hardware the only other physical consideration is the memory mapping. All slave memory is dual ported, shared with either the master or the frame stores/VOS pair. This includes a "window" area used to store sub pictures, which can be accessed via the store/VOS, and a "scratchpad" working area addressable by the master. Furthermore, even the slave's program memory may be commandeered by the master although this usually only occurs when the slave's routine library is booted across at start of the day. Obviously slave buffer control effects the run time activity of the processing i.e. a slave cannot run without some memory. Therefore the boards are rich in buffer modes which are detailed in Appendix (F). By the use of well planned strategies it is

possible to keep the slaves working even whey they are excluded from sections of their memory maps. As a worst case example, a window dump and a master scratchpad access may be in progress, yet even so the slave could still function within program memory to perform some useful task. In fact the only time when a slave is guaranteed unemployed is during boot up.

2.8. MEMORY MAPS

The achieved expanded data memory maps for the master, slave and future sub masters are shown below:

Slave

0000-0FFF	Scratchpad block 1	4K x 16
1000-1FFF	Scratchpad block 2	4K x 16
2000-2FFF	Window memory block 1	4K x 16
3000-3FFF	Window memory block 2	4K x 16

N.B. Slave program memory does not require expanded address.

Master

0000-0FFF	Slave Scratchpad block 1	4K x 16
1000-1FFF	Slave Scratchpad block 2	4K x 16
2000=2FFF	Slave program memory	4K x 16
3000-3FFF	Unused	4K x 16

The master can switch its memory map into any slave address space which it might do for booting up, result passing, or perhaps in an off-line case merely for extra memory.

Sub-Master

0000-0FFF	Scratchpad block 1
1000-1FFF	Scratchpad block 2
2000-2FFF	Window block 1
3000-3FFF	Window block 2
4000-4FFF	Slave Scratchpad block 1
5000-5FFF	Slave Scratchpad block 2
6000-6FFF	Slave program memory
7000-7FFF	

Note: Memory locations are the same with the MSB of the 16 bit address set, however subsequent accesses cause autodecrementing as opposed to autoincrementing addressing.

Deeper trees would in theory speed up a particular process e.g. if a slave becomes a sub master over two new slaves, there would be 200% increase in possible processing power at a node, which with TMS32010s would be a 5 to 15 MIP transition. However, the increased demands on the operating system would tend to degrade this due to the increased communications traffic and data flow. Despite this it is still a worthwhile step to take if a sub task

or tasks are identified for which it is vital that a speed up is achieved. Best efficiency will be achieved for long tasks requiring rare communication, often encountered when transforming picture areas when a large amount of data is involved.

2.9. COMMUNICATIONS PROTOCOL

Communications protocol is kept intentionally simple. Each slave PE requires a control vector from a master to initiate one of a library of sub tasks after which operation is autonomous until end of task is signalled via a priority encoded interrupt and results transferred using the dual ported scratchpad ram. Because of this mode of operation it is possible to use a combination of PE types and hardware within the general framework. The inherent advantage is that new designs may be incorporated without redesigning the whole system, thereby providing a suitable basis for on-going research and development.

2.10. ESSENTIAL INTERSLAVE COMMUNICATION

Many image processing tasks require no essential interslave communications during a particular task if partitioned correctly. By essential it is meant in this case that a task cannot proceed to any sort of conclusion unless a message is received from some external process. In the research although end of task result passing was necessary, result passing within tasks was almost

non-existent. However, this method of communication is supported using slave send and receive buffers administered via software channels in the master, described in chapter 4. A hardware link was impractical due to lack of space on the slave boards, but if this communications mode become of great importance for a sensitive task then a direct linkage or network must be provided. Recent products are encouraging with ethernet and token passing lans being supported by just a few VLSI chips, which might be accommodated on future slave expansion boards. Slaves and master could access the net as a peripheral and have send receive buffers, source and destination address in much the same way as the software method, without the overhead of passing through the master.

To summarise the multiprocessor TMS32010 system is a parallel processing system of high speed DSPs embedded within an expandable architecture. Protocol is sufficiently simple to allow computational units other then TMS32010s to be used and addressing ranges are limited to small window areas. In this manner the addressing overheads for the PE are minimised but necessitate a specialised transfer tool to provide suitable data to the windows. Such a tool would be able to download picture sub areas to the slaves and by using variable resolution and orientation techniques maintains the sequential data streams essential for maximum DSP efficiency. From such needs arose the Vector Orthogonal Scanner described in the following chapter.

CHAPTER 3

THE VECTOR ORTHOGONAL SCANNER (VOS)*

3.1. VECTOR SCANNING

It has been previously mentioned that the window memories associated with slave processors are considerably smaller than the frame stores and that their addressing is optimised for sequential access. This necessitates the use of a hardware memory manager and address manipulator which can effect data reduction whilst maintaining the sequential ordering. Data for analysis could then be transferred from a frame store sub-image to slave memory and furthermore by providing a variable resolution feature, the window may assume greater spatial importance if required. The advantages of such expandable windows are significant allowing tests to be carried out at different scales, e.g. a thin line detector might be used at lower resolution to identify a thicker line, the data transfer time for each dump being identical. Obviously bludgeoning use of pixel and line skipping on raw data causes marked posterization which not only ignores existing information, but more dangerously introduces new "rogue" effects into the data. In applications where this is extreme or when better accuracy is required , the transfer should be preceded by some low pass or averaging filter function, the Gaussian being the ideal (ref. 36).

* This invention is currently undergoing a Patent investigation

If dumping from a predefined image area is achieved via a horizontal line scanning technique (similar to PAL raster) then window data will be optimised for tests on sequential streams in the horizontal direction. However, to support this mode alone is inadequate, as even simple image analysis techniques require tests along other directions, especially vertical and the diagonals. Data can be forced sequential along these orientations by scanning parallel to the required vector (not necessarily horizontal). The geometric importance of this is that the image is apparently rotated, without introducing software overheads and whilst maintaining the desired form of data stream.
To minimise overheads, memory transfers must be very high speed, which if achieved not only solves system memory management problems, but also provides a transform for use in vector scanning algorithms. The physical realisation of this is the Vector-Orthognal Scanner machine (VOS).

3.2. THE VOS MACHINE

The VOS machine is practically two closely coupled circuit boards (ref. 29) the "Frame Addresser" and the "Window Addresser" PICS (15,16) which together have a maximum possible board to board transfer rate of 14MHZx16BITS or 224MBITS/SEC. The former includes the framestore address generator and the transfer driving database. The latter produces the addresses for the slave windows, but also houses the main timing sequencer and control logic.

The main sub sections of the VOS machine can be seen in FIG (17). Before discussing the constituent units, consider the (-)45° vector scan of FIG.(11). The important scan parameters are start address (START), line increment (LINC), newline increment (NLINC), end of line (EOL) and end of dump (EOD). For a regular dump, i.e. one from within a rectangular boundary, all these values are preset constants. As frame store memory is accessed as a continuous 1-D array, (starting top left and working from left to right to the bottom right hand corner), scanning a series of elements is achieved by increments or decrements of an array pointer, relative to an initial reference position. The actual addresses are computed using two accumulators, which operate as shown in FIG. (18).

Basically the dump consists of a line increment scan until a sufficient number have been performed, when a flyback increment occurs relative to the start of the previous line. This process repeats until an end of dump signal is detected. An incrementing technique is used rather than supplying absolute addresses as it requires less storage, i.e. 13 bit increments (max) as opposed to 18 absolute and also it allows spatial relocation merely by changing the start address.

For a variable boundary dump, the driver parameters will vary from line to line, which can represent a substantial amount of data requiring storage. A ram bank is provided for this purpose which allows LINC, NLINC, and EOL to be different for consecutive lines and also provides the EOD flag. For every line transfered, two layers of ram are used, configured as shown in FIG. (19) the top one containing LINC, EOL and EOD, the bottom is similar, but has NLINC instead of LINC. Provision was made for 512 simple lines within a 1024*24 bit stack of 35nS ram, the levels of which were cycled through during a transfer. FIG. (20) shows the difference between a simple and a multiple line. The latter is merely treated as two or more simple lines, the only disadvantage being that more stack levels are required. Dump termination occurs when the EOD output of the ram becomes active.

Generating driver parameters for variable boundary shapes takes time, which can be much reduced by adopting square dump areas where acceptable. When a regular boundary is used there is no need to fill the driver stack so deeply and in fact only the top two levels are required. The problem is then how to signal EOD. This is solved by including an auxillary EOD mechanism held in PROM, from which one of several common EOD values may be software selected.

3.3. FRAME STORE ADDRESSER (FADD)

The block diagram of FIG (21) shows the essential elements of this address generator, including the two accumulators introduced earlier. ACC1 is responsible for maintaining a running address count along any particular line. ACC2 contains either the initial start address or that of the preceding line. The master buffers are only enabled for initialisation to copy start to ACC1 and ACC2, which during run time have their inputs driven from the feedback buffer on ACC1's output. The increment values are latched prior to input in ACC1 to ensure their presence for almost a full clock cycle. The circuitry is fully pipelined and can latch out addresses at the maximum rate of 14MHz. These can then be driven to one of the two frame stores P&Q along with a valid memory address strobe and a R/W line, to allow not only dumps to, but also loads from window memories.

At first glance FIG. (21) might suggest that the main accumulators work independently of EOL and EOD. Actually these flags are vitally important to FADD operation which they influence not directly but via their effects on the timing signal controller described later.

Under some circumstances, address increments are unsuitable and decrements required. For this purpose a negative jump capability is provided by modifying the MSBS of the increment bus and

performing one's complement subtraction. Whether in decrementing or incrementing mode, addresses for the slave window areas also need to be generated to provide a sink or source for data.

3.4. WINDOWS

Each slave window consists of a 8Kx16 block made up from 2x4K blocks, plus associated buffers and timing. The memory is basically a 0-8191 array which can sit on one of two window address busses S or T. These busses are addressed by either the Normal Scanner or the Orthogonal Scanner.

3.5. NORMAL SCANNER

In normal block transfer mode a counter is used to generate window addresses, cleared on initialisation and clocked by window valid memory address strobe. The count produced can then be driven onto the T & S busses as required, however, due to strict timing constraints the MSB of the address (4K block select) is produced in both normal and inverse form and bypasses the final output stage to provide a time advance relative to the LSBS.

Strange as it might seem, there are occasions when the normal scanner is required to be preset to a negative number (see later Chapters) prior to a transfer and therefore this facility is available under software control.

3.6. ORTHOGONAL SCANNER

During the early design phase of VOS it became apparent that it was possible to produce a 90° clockwise rotation of the dump/load image simultaneously with the normal scan. A proviso was that the address machine must work in fixed boundary mode, giving a known and constant line length necessary for formatting the memory. A simple example is given below, which demonstrates the procedure for a horizontal scan also generating a vertical dump.

(LINE LENGTH = 3 NUMBER OF LINES = 3)

	| 0 1 2 |	WINDOW
FRAME STORE	| 3 4 5 |	ADDRESSES
	| 6 7 8 |	

X X X X X X X X	STAGE 1	STAGE 2	STAGE 3	
X X X X X X X X	ABC	ABC	ABC	NORMAL
X X|A B C|X X X	XXX	DEF	DEF	ADDRESS
X X|D E F|X X X	XXX	XXX	GHI	
X X|G H I|X X X				
X X X X X X X X				
X X X X X X X X	XXA	XDA	GDA	ROTATED
	XXB	XEB	HEB	ADDRESS
	XXC	XFC	IFC	

The hardware to produce this rotation is shown in FIG (22). The procedure for the simple example is described below:

i Clear latch, load down counter with line length minus one (2), load offset latch with line length (3), set MUX to select counter.

ii First line address becomes first pixel in last column i.e. 02 instead of 00.

iii MUX selects offset latch and for next pixel line length (3) is added to previous address, i.e. 02+03=05 therefore second pixel of first line becomes second pixel of last column.

iv Line length is added again to give 05+03=08 i.e. third pixel of first line becomes 3rd pixel in last column.

v At end of line the counter is decremented, and reselected i.e. 02-1=01.

vi First pixel of second line becomes first pixel of second column 01.

vii MUX selects offset latch and line length added to addres 01+3=4.

and so forth.

Current implementation allows for two 'T' slaves and two 'S' slaves so called by virtue of their window bus connections. The driving of these busses is totally flexible, i.e. both could be normal or orthogonal or one of each. Producing say a horizontal and vertical scan simultaneously is useful for finding central co-ordinates where both slaves run the same process, but because of the data rotation generate both the X and Y co-ordinates. It is also worth noting that any number of slaves may "listen" in to a dump broadcast for perhaps applying different tests to the same sub area.

3.7. FINITE BIT MAPPED DISPLAYS

The discussions of vector scanning given thus far have ignored the effects of addressing finite bit mapped pictures. In fact for a 384x577 screen with 4/3 aspect ratio only the following scan vectors are advisable for block transfer:

0° 90° 45° 26.6° 56.3°

This is because block mode only addresses pixels which lie exactly on the specified vector and therefore certain angles have large interpixel distances. Weighting coefficients for the above angles can be normalised with respect to the horizontal interpixel distance to give:

1 0.5 1.41 1.12 1.8

The higher the number the greater the spatial significance of the pixel, which may be considered as a reduction in sample rate.

Enhancements, such as filtering or edge detection for example, would probably only require a subset of the above angles and would therefore be implemented using block mode. If the 45 (or -45) degree scan is considered, the associated resolution reduction leaves holes in the area scanned out. This can be overcome and the display maintained by one of two methods:

a) the test is performed then redone from slightly different start points effectively interleaving several block scans
b) as (a) but the test performed on one block and then duplicated in the others.

Mode (a) makes better use of the available information, but (b) is much quicker. In fact the second mode forms the basis of a hardware zoom function where small areas can be blown up by pixel and line duplication. This is a reverse effect to resolution reduction mentioned earlier, but of course, no new information is added. With suitable filtering a zoomed sub image may be more conveniently measured at the new scale.

3.8. VECTOR LINE SCANNING

There are occasions when vector lines are required which do not exist in the block mode set and which could not tolerate large interpixel weighting coefficients. The solution is not immediately obvious especially when considering that the frame addresser must access physical pixels with a regular ratio of X and Y increments. An accurate method of interpolation would calculate the contribution of each pixel fraction covered by the line, to estimate the required pixel, however the address generation for this would be very complex and several pixels would need to be addressed just to produce one scan point. A more practical engineering approach makes use of the VOS machine in line as opposed to block mode.

It is possible to represent a vector by a number of small shifted lines as shown in FIG. (23). Such a pattern can be traced by treating each small line segment as a complete simple line and the whole vector as a multiple line, i.e. collection of simple lines. By choosing suitable ratios it is possible to generate scans which on average represent the scan line required. Use of these modified horizontal and vertical scans greatly reduces interpixel distance, but at the expense of introducing a displacement error (De).

This error must be minimised as it is not only a measure of departure from the ideal vector, but also of the non-linearity of sampling along that direction. It was found that by careful choice of ratios that angles could be defined to within a degree [Appendix (G)] and the error kept within a 1.5 range relative to horizontal interpixel distance. Note that interpixel distance for all line scans is unity.

Vector line scanning is used extensively for the perimeter coding technique in Chap [7] for which 32 different scan lines are required. Sample vectors are shown in Fig [24]. The unshaded sectors can all be achieved using increments and starting scans from the top most and then left most pixel, which is the usual mode of scanning. The shaded sector is a special case of modified horizontal scans which require decrements and that scans start from the bottom most and then left most pixel. For this type of scan, to start from top left would require that the scan pointer pass from right to left under the control of the LINC constant. Whilst this is practical in terms of circuit implementation it violates the definition of Pseudo Random Access (Chap 1) for which the frame stores were designed.

Errors would result because instead of staying within current chip address space or moving to the next consecutive one, there could be accesses to previous un-powered rams. For this reason the special case line scans are started from bottom left positions in

which case LINCS cause left to right movement which is consistent with Pseudo Random Access. The decrements occur for NLINCS and are achieved using 1's complement subtraction. The extra wait state provided for simple line flyback allows the state machine predictors on the frame stores to stablise after a NLINC jump whether positive or negative so no errors are introduced.

Due to the higher ratio of NLINCS to LINCS vector line scans are usually slower than block transfers on a pixel by pixel basis. However, as a line of information contains fewer pixels than even modestly sized sub areas, line dump times are quicker overall.

3.9. THE STATE MACHINE DEVELOPMENT UNIT

Obviously to control such a large, fast synchronous machine such as the VOS with its many feedback paths and numerous control lines necessitates the use of a high speed timing controller. This presented a major problem, because to produce the 14MHz address strobes required that any synchronous state machine used for timing would have to be clocked at twice this rate, i.e. 28MHz. At the time of design it was difficult to find a logic/latch combination that could achieve the 35nS cycle required. Allowing 10nS for set up and clock to output of a high quality latch (74 ADVANCED SCHOTTKEY) left only 25nS for the feedback logic. Several alternatives were considered:

a) Discrete (ref. 37)

b) ECL (ref. 38)

c) Small bipolar PROMS (ref. 39)

d) Pals (ref. 40)

e) RAMS (ref. 41)

It soon became apparent that a) was impracticable as the state feedback even after Quine McCluskey minimisation was much too complex to be achieved with simple gates. Using small proms was also of little use, because although they could just achieve 25nS the delay of the paging logic used to control the chip selects had to be added to this. In terms of technical specification both ECL and high speed Pal devices seemed attractive. ECL is not however TTL compatible and requires extra conversion I.C.'s all of which run from an unusual supply voltage and require special circuit layouts. Coupled with the high cost, low availability and the need for programming by a third party, ECL was discounted. The Pals were more encouraging, they could just achieve the required speed and could accommodate the feedback logic despite their limited number of prime implicants. A module was constructed to realise the controller in Pals as a plug-in to the main board. Whilst it was feasible to populate the board and commission it, there was a major practical drawback. The Pals were expensive, especially when considering the cost of third party programming and to assume that the state machine characteristics would be fixed and error free throughout the development was somewhat naive. So although the Pal module was recognised as a requirement in the distant future for specific applications, some interim solution was required to allow convenient and inexpensive

modifications to the state machine. From this requirement evolved the State Machine Development Unit (SMDU) which is a 25nS ram based synchronous state machine as shown in FIG. (25).

The principle parts of the state machine are the primary latch, the feedback ram, the two output decoder rams and the secondary latch. The secondary latch is necessary to eliminate glitches on ram address transition and to maintain output levels fixed for complete clock cycles, in this way the outputs from the SMDU leave in a synchronised fashion, i.e. having suffered equal propogation delays. The machine responds to a user applied start strobe to begin a transfer and then works autonomously generating clocks and strobes, detecting new line sequences via the EOL signal. Operation is terminated when external EOD circuitry inhibits the start flag. The state diagram for current implementation is mapped in FIG (26) along with some signal description. Because the transfers are pipelined the valid memory address strobes for the framestores and windows must be phased relative to each other. This complicates the sequencer requiring post valid memory clock (POSTVMACLK) for dumps and pre valid memory clock (PREVMACLK) for loads as well as the usual strobes. A further complication arises from the sluggish nature of one of the inputs namely EOL. The circuitry which produces this flag does so by comparing the running line count with a fixed constant until some transition is reached. However, the time delay is such that the flag will not effect the state machine until one or more clock cycles have passed. To overcome this EOL must be phase advanced,

achieved by reducing the EOL constant. This is sufficient for fixed speed operation, but it is very important to appreciate that if the pulse clock frequency is varied, the number of cycles of advance required may change. Without correction a circuit which works at high speed may not work at reduced rate, which might be baffling to the uninitiated. By a similar argument the single step function (described later) must be used with caution.

On power up ram will contain undefined values which must therefore be overwritten by intelligent data. The microcode to be installed is held in EPROM. The mechanism for booting to ram involved a counter some multiplexers and a Pal device. All that is required from a software initialiser is to pull one of the Pals inputs low for a short time during which, addresses, chip selects and write clocks are generated to effect the boot. Note that this Pal is inexpensive and easily programmed as opposed to those intended for the Pal module timing controller. It may well be that the state machine characteristics are required in several forms and these could also be accommodated in the EPROM and selected using the MSBS manually or via software.

The hardware mentioned is not just the minimum to achieve the state machine, but provides extra facilities to help develop the machine and hence the name State Machine Development Unit. Monitoring the action of the SMDU is a good way of assessing the "health" of the VOS, for this purpose a bar graph is provided on WADD which displays current outputs, with auxilliary L.E.D.S.

displaying via monoslables pulse clock (RED) and EOL strobe (GREEN). with some experience run time light patterns can be identified as good or bad or alternatively a single step facility can be used with care to cycle through the states. It is also possible to read back the ram contents using the single step function and the bar graph output. The debug modes of operation are controlled manually by a number of switches selected as shown in Appendix [H].

3.10 HIGH SPEED DESIGN CONSIDERATIONS

The need for high speed circuitry to perform identified hardware procedures has already been discussed, but as yet no mention has been made of the considerations and precautions necessary to ensure the integrity of signal transmission (ref. 42).
Pipelineing is recognised as an invaluable means of "accelerating" electronic processes. Its implementation usually requires that propogation delays be equalised rather than kept to an absolute minimum, providing of course that they satisfy a specified lower limit of throughput rate. If a delay is accurately known, it can be compensated for, so it is only uncertainty in delay values that tend to limit system cycle times. Usually component manufacturers specify upper and lower bounds on propogation delay as the actual value is likely to vary with device and temperature. For a robust design uncertain delays must be assumed to act in the most disadvantageous fashion.

Another source of uncertainty is due to wiring interconnect delays which are very difficult to predict, a rough rule of thumb being 4ns/foot for wirewrap wire. The constraint on the VOS machine was that rams rated to 45nS could only achieve a maximum cycle rate of 70nS due to 25nS uncertainty in transmission. At this rate of transfer signal wires begin to behave like radio transmitters and receivers causing crosstalk. On wire wrapped boards this effect can be made less severe by adopting direct point to point wiring, which results in such random effects that serious crosstalk is avoided although some noise is introduced. A common problem occurs when busses are wired "neatly", when the adjacency of many parallel conductors is tantamount to encouraging crosstalk.

During board to board transmission many signals must run parallel to form a bus so special precautions are required. Where ribbon cable is used, signal wires should be interspersed with ground lines to reduce local interference and in addition an earthed screen should be provided around the cable to prevent interference to or from the group as a whole. If separate wires are used, they should form twisted pairs with ground lines so that induced currents in both send and return lines tend to cancel out. Having taken such precautions it must be appreciated that the A.C. impedance to ground of the transmission lines is reduced (approx 50 to 150 ohms for wire wrap) which therefore demands higher performance bus drivers than would otherwise be necessary.

The above precautions and considerations were to minimise the window dump overheads to the level where they were negligible with respect to even the most modest software process on the dumped data. In achieving this, a very high speed machine was realised, the capabilities of which were not exhausted by the transfer alone.

3.11 THE DATA TRANSFER PROCESSOR

In a conventional computer, access to external memory involves an I/O time penalty. VOSTTAC1 is no exception and in fact with the large address space associated with a TV image it is very sensitive to this effect and hence the high speed of the VOS. However, when using this machine it is not really fair to label all transfer time as overhead, especially when the geometric rotations, windowing and resolution skipping save a great many software instructions.

By installing higher level processes into the dump mechanism, transfers may be considered not only as an overhead to be used sparingly, but alternatively as a very high speed process to be employed extensively. The VOS can be considered as an efficient low level signal processor when used in conjunction with Data Transfer Processors (DTPs). These are special hardware units built on to each frame store card to enable software selected data

processes to be applied when some form of transfer is occurring. The block diagram of FIG (27) shows the three sub sections of the current DTP implementation comprising the Time Shift Subtractor (TSS), the Convolver (CNV), and the Multi Dimensional Thresholder (MDT). The last two are used exclusively by the VOS with the convolver only being possible due to the sequential data stream generation. The TSS also requires flowing data but only when initiated via the video address scanner. The use of the DTP is discussed in more detail in later chapters, but its close association with the VOS machine required that it be introduced here.

To summarise then, the Vector Orthogonal Scanner is a high speed memory transfer tool, capable of transferring irregular or regular bounded sub images to or from slave windows with optional reduced resolution. Furthermore the dump/load may proceed by scanning parallel to vectors not necessarily horizontal. By constraining the area of interest to be square, not only can the vector dump be produced but also an additional dump apparently derived from a scan at 90° to the actual one. All complex timing control is handled by a high speed ram based timing sequencer, to perform fully pipelined board to board transfers at a maximum rate of 0.224 GBITS/SEC. Used in conjunction with DTPs the VOS becomes a very high speed low level signal processor in its own right.

Having designed and constructed a complex piece of hardware it is necessary to be able to control it in an efficient manner, utilising the many facilities during program execution. The driver signals and software for the VOS and DTP are discussed in the following chapter along with all other system driver and utility programs.

CHAPTER 4

SYSTEM SOFTWARE

4.1. SOFTWARE APPROACH

Anybody who has ever considered buying a home computer would know that no matter how innovative or highly specified the hardware a machine would sink or swim by virtue of its software support.

VOSTTAC1 is a collection of specialised electronic machines which need user programming to achieve co-ordinated action. Although speed considerations demand closely coupled hardware and software, sufficient flexibility must be retained to allow general use. The proposed solution was a suite of driver and utility routines, written in machine code for speed, but with modular construction and convenient parameter passing for ease of use.

In its fastest and most basic sense, a real time operating system might be considered as a number transmitter and receiver synchronized to a time sequence of events. Whereas computers are quite contented when working with only number traffic, humans in general are not happily employed with such traffic control. More user friendly systems offer higher levels of abstraction to the extent that the user is totally isolated from the machine characteristics, especially when high level languages are employed. Maintaining this cosy programmers world would be very time consuming and dissipate the possible impact of specialised

high speed equipment. Such a concept is totally alien to the design philosophy of VOSTTAC1 and therefore a lower level system was sought which could use any available specialist functions/processes in an efficient manner. The current version of the system achieves this yet provides a level of symbolism within structured program modules. A necessary requirement for any would-be programmer is therefore familiarity with the logically defined modes of hardware operation, but not necessarily the in depth sequences of control which implement them.

4.2. MASTER CONTROL

In Chapter 2 it was mentioned that VOSTTAC1 contains a tree structure, perched on top of which is the master processor, responsible for the supervision of all units on the tree branches immediately below it. In the current configuration this lower level pays host to the video scanner, the VOS machine, the DTPs and up to 4 slave processors.

On an offshoot of the tree sit the master peripherals (always connected before the expansion board) which may include the ADC/DAC/Nomination rig and BBC model B screen dump interface adaptor. It is worth noting that any future sub masters would only be responsible for their peripherals and slaves and not the system hardware utilities on level 2. To produce a concerted effort from all these different sub-systems requires master software which supports the following:

a) Real world data I/O

b) I/O to control registers and flags on hardware units

c) Safe initialisation of all hardware to desired function

d) Control of all buffer modes and multiplexing

e) Recognition of different events via interrupts

f) Initiating tasks and logging their completion

g) Controlling shared memories

h) Effecting data transfers

i) Synchronisation to external video

j) Result Display

NB: Not necessarily in order or importance.

4.3. I/O CAPABILITIES

A first step to realising any of the above functions is to consider the external I/O options open to the master, which exist on several levels (ref. 43).

LEVEL	DESCRIPTION	CAPABILITY
A	Primary Ports and Interrupts	I/O
B	Expanded Ports	I/O
C	Expanded Memory Address	I/O
D	Priority Coded Interrupts	I

Level A is the most direct and its port map was discussed in Chapter 2. In fact it is through this mode that BC&D are implemented and then used to control the main systems of VOSTTAC1. Each board or sub-unit has one or more expanded ports dedicated to control or flagging allocated as per table.

EXP 0	Slave 0	Cntrl	EXP8	Video	Cntrl Low
EXP 1	Slave 1	Cntrl	EXP9	VOS	Cntrl
EXP 2	Slave 2	Cntrl	EXP10	VOS	Line Offset
EXP 3	Slave 3	Cntrl	EXP11	VOS	Line Length
EXP 4	DTP	Cntrl	EXP12	VOS	Start High
EXP 5	DTP	Arg	EXP13	VOS	Start Low
EXP 6	STORE	Flag	EXP14	VOS	Ram High
EXP 7	Video	Cntrl High	EXP15	VOS	Ram Low

To activate this control mechanism requires the use of two primary ports, OPSTRM & STREAM. The bare minimum required to effect an information transfer would be to output an expanded port code on OPSTRM, to open a communications stream between the unit and master and then transmit/receive the data down STREAM. However the required expanded address word is not meaningful to a programmer and tedious to calculate so board I/O is dealt with via logical unit numbers instead. The code conversion is performed by the utility routine LUNAK. Although there is usually one to one

correspondence between expanded ports and logical unit numbers, there are special LUNs which can access groups by opening multiple streams which is a very useful property when broadcasting similar controls to several slaves.

Mode C access was introduced in Chapter 2 as a means of reading from or writing to slave scratchpad or program memory by the master using the MXDAT channel. The memory pointer to which MXDAT refers can be set by writing to MXADD which not only defines the current address but also selects autoincrement or decrement addressing mode depending on the MSB (1 = decrement). However this alone is insufficient to access slave memory as prior to a data transfer a control code must be issued to the slave (or slaves) to force the required buffer mode. Usually only one processor memory is 'paged in' at any given time although group processes are possible with care.

Master peripherals are controlled exclusively by Mode A, i.e. employing dedicated primary ports. The main reason behind this is to preserve compatibility with slave peripherals which without expansion do not have any other facility for data I/O. As a consequence master and slave peripherals may be interchanged with only minor modifications. Obviously the type and complexity of a particular peripheral will largely dictate the actual port usage. A current example is the ADC/DAC/NOMINATION RIG. This peripheral

is fairly complex, containing a Z80 processor and shared memory, which infact requires more I/O than could be accommodated on the available primary ports. To overcome this problem, advantage is taken of the fact that the unit works primarily with 8 bit values as opposed to the 16 bits of the TMS32010. It was therefore acceptable to use the upper 4 MSBS of the data as an auxilliary port decode, peculiar to the peripheral. More information is given about this rig in the driver section.

Mode D is amongst other things the answer back or handshake to any request made by the master, for example a slave signalling the completion of some task. In addition this mode is also a means of flaging, whereby free running real time events may be acknowledged. Practically this is achieved by using priority coded soft interupts (BIOs). Any level of interupt will generate an active BIO, so to identify the source the priority latch (PRIOTY) must be read to find the most significant event. Eight levels are supported reserved as shown below,

BIO	0	Frame Start Flag
	1	VOS
	2	Slave 0
	3	Slave 1
	4	Slave 2
	5	Slave 3
	6	Timer
	7	Manual Abort

N.B. 7 = Most significant BIO

4.4. DRIVER EXAMPLES

Using the various levels of master I/O it is possible to create software drivers which enable all the hardware functions to be utilised. These utilities and in fact all other master programs make use of variables and constants held in internal datamemory, which for the TMS32010 consists of 128 words in page 0 and 16 words in page 1. The partitioning of this memory is shown below:

Location	Description
0 - 2	Simple Constants (Zero, One, Minus)
3 - 15	Temporary Parameter Block (P0 to P12)
16 - 63	System Global Constants and Variables
64 - 127	Temporary Data Buffer
128 - 143	Slave Boot Variables/Constants

To overcome restrictions due to the small size of internal datamemory some of the global parameters and "boot" variables can be overlaid with other values held in program memory. The temporary parameter block is used primarily for value or result passing between system software routines. Alternatively this area may be used for temporary or local variable storage as can the data buffer, although the latter is generally employed to interface with external datamemory.

Some of the globals have already been introduced by name in earlier chapters but a fairly comprehensive list is given in Appendix (I). The task of any driver program would be to use some of these values in conjunction with temporary entry conditions to effect some change or action in physical units connected to the master. A number of the more useful drivers are described below.

4.5. VIDEO BOARD CONTROLLER - VIDEOZ

VIDEOZ is an important routine, not only does it enable initialisation of the video sections, but also provides a mechanism for software synchronism to external timing signals. It is often used in the following manner, either directly or within higher level programs.

1) Initialise video board and enable toggle and interrupts
2) Observe interrupt generated by start of new frame, then disable the toggle and interrupts.
3) Begin main process
   (N frame times required)
4) Re-enable toggle and interupts
5) Wait for interrupt then go to 2 (and repeat)

This procedure ensures that picture flow through the system is maintained in such a way that data is available for processing for as long as necessary. The inputs for VIDEOZ are discussed below.

P2 This selects one of several video sources to be used by providing a selection code, i.e.

1 - horizontal test bar picture

2 - programmed number of fixed colour screen

3 - analog to digital converters

P3 This is the number for P2(2)

P4 As the system can run from either external or internally generated timing signals, this parameter selects between the two, i.e.

0 = external

1 = internal

P5 Under free run conditions the frame stores will toggle between slave and video access on every frame. If uncontrolled this is likely to corrupt processing that requires longer than one frame period and so a toggle inhibit is provided. To allow synchronism of the software with frame timing a soft interupt (BIO) line is driven to the master to indicate start of frames. The function of the P5 input is to enable or disable the toggle and soft interrupt

1 - Toggle

0 - Disabled

P6 When updating a frame store, it is standard practise on VOSTTAC1 to perform read before write cycles in which case the processed contents of a slave are displayed and not the new overwritten data. However there are occasions, perhaps when store contents are jumbled when it is more meaningful to display incoming data and therefore a write before read mode is supported selected by P6.

1 = read before write

0 = write before read

P7 If P5 is used to disable the toggle it also disables the BIO line. When using automatic background subtraction it is necessary to have the toggle disabled yet allow BIOs to flag new frames, therefore an auxilliary BIO clear is provided by P7

0 = auxiliary BIO clear

1 = enable

If VIDEOZ can provide the data flow for the processor it is then up to the master to initialise them for co-ordinated action. What better way to organise a group of slaves than to delegate to a slave driver!

4.6. SLAVE DRIVERS

To employ a group of slaves to achieve desired goals, requires some initialisation and knowledge transfer. But a prerequisite to this is that the number of available slave processors be known.

In the context of an open rack based computer system this may seem rather trivial as the number of processor boards can clearly be seen. However by measuring the processing "resource" via software it is possible to select one of the options of several different strategies for the available slave configuration. Furthermore should processors be improperly installed or malfunction, the software would automatically adapt to achieve the designated task, a property which is very useful for fault tolerant rerouting. The routine for actually assessing the processor resource is, RSCFND which from a knowledge of possible slave addresses and associated interupt levels (held in tables LUNSYS and FBIOTB) attempts to test shared memory spaces to find physical processor boards. Whilst identifying those present it compiles two new tables SLVTAB and SLBTAB which contain valid slave addresses and associated BIOs. Once these are complete the processors may be accessed via logical slave numbers (LSNs) which are a further abstraction from the logical unit numbers previously mentioned. For example a one slave sequence would always use LSN0 irrespective of the actual address or physical position. Similarly a two slave sequence would use LSN0 and LSN1 and so forth.

Having identified the useable complement of processor boards, whether by adaptive software or operater assumption, the next step is to "boot" them up, i.e. install all the required machine code in the program rams. Initially these slave routines are held within the master program and then transferred across by a

bootloader during initialisation. If RSCE is used prior to this, all that is required to effect the boot is to call BOOTEM. This is a higher level function built using the primary driver SYBOOT, the latter being discussed below.

During initialisation the internal datamemory page 1 is reserved exclusively for use by SYBOOT and allocated thus:

143 AARVCO Jump Code for Reset (F900)

142 AARVC1 Jump Address for Reset

141 AAIVCO Jump Code for Hard Int. (F900)

140 AAIVC1 Jump address for Hard Int

139 AASEL Dump Start Pointer (Usually Syboot)

138 AACUT Dump End Pointer (Usually Syfin)

137 AAUSCH Slave Address

136 AAVECS Transfer Mode Select

1 = Transfer vectors and Prog.

0 = Transfer Just the Prog

135 Run Time Variables

to

128

Any corruption produced by SYBOOT will change slave programs and very likely cause disasterous effects when run as part of a main system program. Because of the vital importance of program

integrity, every transfer is automatically verified by reading back after writes and any failures are logged in the variable ERFLG1. On completion of a boot, the relevant slave is reset to initialise it and the generation of an answer back BIO a short time later in conjunction with zero errors is usually sufficient confirmation of a healthy boot and processor. If SYBOOT or any other relevant programs are immediately followed by a call to ERTRAP, then ERFLG1 is examined and if non zero, causes one of several error messages to appear on the auxilliary display. This facility is unlikely to be needed with SYBOOT unless VOSTTAC1 is in close proximity to several arc welders! The error display mechanism is more useful for highlighting logical programming errors when things are not going quite to plan.

The programs that are actually transfered to the slave or slaves follow immediately after the boot loader in master program memory space, the last of which is followed by the SYFIN end marker. In fact not only are these slave routines transfered but SYBOOT as well. This self booting operation is necessary, should the tree structure be extended downwards in which case the boot could ripple down the tree through various sub masters.

After booting up a group of slaves it is then necessary to control them during program execution. An important function is to ensure that their memory buffer modes are suitable for the given operation: changing the modes is most easily accomplished by using SLMODE where the inputs are.

P0 - LSN (Logical Slave Number 0-3)

P1 - Buffer Mode Code

During the course of an application program the master is likely to require several different sub tasks to be performed by a slave. To actually run a slave process the master generates a hard interrupt vector, which triggers a period of autonomous slave action, the completion of which is signaled via a soft interupt handshake. Normally to initiate a task, VEC320 is employed where the inputs,

ACC = LSN

P2 = Vectored interrupt

The stream down which the interrupt is performed is assumed open. The event is recorded within a communications register which is described later when master priority interupt handling is discussed. A special case slave interrupt routine is CLMBIO which is a mnemonic for the utility program which clears the Master BIO line generated by a slave. All it requires to operate is an open stream, however by contrast with other vectored interupt routines it does not generate an answer back. Obviously there would be little merit in a program which cleared the BIO line only to immediately set it again.

At the slave end of events, vectored interupts can be differentiated between by reading the 8 MSBS of GETVEC. This value is then used as an offset to read SVECTB, which is a table containing the start address of all the service routines. Program flow then branches to the designated area, until completed, when outputting the contents of ZERO on primary port MBIO flags the master.

At sometime during the course of a slave process it will be necessary for picture information to be transfered between the frame stores and windows. This requires action from the VOS machine, which although its capabilities and advantages have been already expounded, its software control has as yet been conveniently overlooked.

4.7. VOS DRIVERS

Before the VOS can be driven in anyway it must be recalled that the heart of the machine development unit is not configured on power up, being ram based. This is easily remedied by calling SMDUBT which controls the EPROM to RAM boot and then exits after a delay leaving the address machine raring to go.

The VOS can be programmed in so many different ways to achieve various modes of operation that it is almost impossible to produce a totally general purpose driver. Having said that there is one

utility program that is used in roughly 95% of applications, namely DRIVAD. This is a fixed boundary dumper/loader responding to several inputs.

P2 Dumpbar/Upload

P3 S&T Bus Codes

P4 EOD Prom Select Codes

EOL Line size of Window

LINC Line Increment

NLINC New Line Increment

STRLLO Absolute X Screen Pointer

STRLHI Absolute Y Screen Pointer

Basically the program forces the hardware to generate the required scan pattern by installing EOL, LINC, NLINC. Start is also used but after calculating the 18-bit screen array address from the X&Y pointers. The role of the local parameters P3 and P4 are not immediately obvious and merit some explanation.

It was mentioned in Chapter 3 that the routing of Normally or Orthogonally scanned window addresses was totally flexible and therefore some mode selection is required. Specified by P3 the modes can be:

MODE	S	T
0	NORMAL	NORMAL
1	NORMAL	ORTHO
2	ORTHO	NORMAL
3	ORTHO	ORTHO

(The Ortho scans only apply for fixed boundary "square" scans).

When using fixed boundary scans it is not possible to use the VOS ram bank to generate the EOD flag and so a PROM mechanism is used. P4 selects the number of lines to be transfered from a subset held in the EOD PROM. There are eight values arranged as two pages of four, which for version
MkV gives:

Page 0	No. of Lines	Page 1	No. of Lines
0	90	40H	89
1	64	41H	63
2	14	42H	13
3	22	43H	21

It is no coincidence that Page 1 values are 1 less than those of Page 0, a requirement for some DTP actions (see later chapters). Calling DRIVAD automatically starts a transfer, the completion of which is signaled by a soft interrupt to the master, in much the same way as a slave process. Once acknowledged, ADDRST is the VOS equivalent to CLMBIO.

DRIVAD is suitable for most general purpose transfers, but for more skilled use of the VOS machine, specialist drivers must be written. A good example of such a routine is that which produces the vector line scans required for the perimeter coding technique in Chapter 7. The most notable difference to DRIVAD is that LINC, EOL & NLINC produce scan lines through a central point rather than a block transfer. In addition EOD is ram based, requiring deeper stack filling, and the routine is designed to run more efficiently for the given application. All that is required to produce one of the subset of 32 line scans is to put the scan vector number (shifted by two) into VCOUNT and call GETVSC providing that VCINIT has been called at some earlier stage.

Two routines worthy of note are MOVWIN and GETWIN. Both of them employ DRIVAD and perform dump/loads using slave windows, but automatically copy it to or from the scratchpad memory. For preliminary investigations this is most useful as it allows the master apparant direct address to window data, an access not

normally afforded. Once the essentials of the VOS machine are grasped, DRIVAD can be used to produce many different types of dump and perhaps less obviously allow display of non picture data at various scales and positions.

In Chapter 3 the idea was put forward that the VOS could be a processor and not just a transfer tool when used with the DTPS. Because of the large number of possible variables when using DTP a general purpose driver would be very unwieldy, it being much easier to generate the control words directly.Therefore rather than explain the DTP at this stage it is better distributed throughout the following chapters as its various sub-sections are introduced.

4.8. ADC/DAC NOMINATION RIG DRIVER

This provides rather a good example of a peripheral driver as it is more complex than most. The ADC/DAC part of the circuit (ref.35) existed prior to the nomination rig and it was the need to access analog input and output channels that necessitated the auxilliary port decode. Only one port is needed as the 4 MSBs of the data provide a 1-16 port expansion, satisfying analog I/O requirements and providing spare capacity for expansion. When the nomination rig (ref.28) was installed, control became more complicated. Central to the rig is a bit mapped video display supervised by a Z80 processor. The processor also uses a 4KX16 ram block shared with the master, for stacking and parameter

passing. The user program must therefore be capable of running tasks on the Z80 (ref.44) and accessing shared memory. A typical task for the nomination rig is to place a confining box about a target and make the top most left co-ordinate of the box (in Grid co-ordinates) available to the master. These values after conversion to absolute co-ordinatres are often used to reference a VOS dump from the area. The routine responsible for this is called FSTNOM which procedes as follows:

1) CALL ZHLTOK - Check Z80 in halt state
2) CALL VECZ80 - Give appropriate vectored interrupt to NOMRIG
3) DELAY
4) If Halt inactive (Running) then go to 5 else go to 1
5) CALL BRQZ80 - Request Z80 bus and wait for BUSACK
6) Read in Start Parameters
7) Remove BRQZ80
8) Wait for BUSACK false
9) Convert Co-ordinates

The relevant routines are:

ZHLTOK	Waits until Z80 halted and then returns
VECZ80	Gives vectored priority interrupt to Z80 (Input PO = Vectored interrupt number)
BRQZ80	Either gains access to Z80 shared memory and sets memory pointer, or returns memory to the Z80

INPUTS:

P5 Memory pointer

P6 BRQ (0 = RETURN, 1 = REQUEST)

The protocol used is not the bare minimum required but as the peripheral is physically distanced from the main rig, the extra precautions were employed to ensure more robust communications.

Due to space limitations only a very small fraction of the many available routines have been mentioned. A more comprehensive list is provided in Appendix (I) along with some brief functional description. For further details the VOSTTAC 1 application program source listing should be consulted (ref.45), the programs within being fairly well commented.

4.9. SEQUENTIAL AND PARALLEL TASK HANDLING

The main unit drivers of VOSTTAC1 flag the completion of their alloted tasks by priority coded soft interrupts to the master. If sequential operation is acceptable, i.e. one task completes before another can start, then BIO line control is fairly simple. Having initiated a task the master then waits for a BIO to be generated, irrespective of level, to signal completion. All that is then required is that the BIO be cleared before the next task is run.

In this mode free running interupts such as those produced by the video scanner should be disabled and store control should be under manual supervision. As this is only likely to occur during preliminary investigations when speed is not of tantamount importance, this restriction is acceptable.

The hardware of VOSTTAC1 is inherently suited to parallelism by design, so to operate it in sequential fashion is hopelessly inefficient when applied to real time operation. Analogies are innumerable but anyone who has ever wasted their lunch hour in a bank queue would have come to the conclusion that more cashiers operating in parallel could increase throughput (perhaps someone should tell the Managers). In terms of software the transition from sequential to parallel processing is by no means trivial. Suppose that at time TØ, N processes with different execution times are started and some time later T1, a particular task is required finished, then some flag must be tested by the master. If the BIO line is tested and found active then at least one task has terminated. By then reading the priority latch the identifier of the most significant completed event can be found. If however there exists a finished task or tasks of higher priority than that under test, it would not be possible to reach any conclusion without first removing the more significant BIOs. Merely clearing these interrupts is insufficient as at some time later the status of the high priority tasks will also need to be considered. Therefore some record of active and terminated tasks must be

maintained and specific tests must be able to strip off higher priority BIOs by first servicing them and making appropriate alterations to task status.

The current mechanism employed has passed through several phases of development and perhaps unusually has become simpler and more straightforward at each stage. Early approaches used quite large task status tables holding the addresses of all driver routines along with flags to indicate running, terminated or error status. Whilst this had a diagnostic feel about it, the table maintenance was rather cumbersome and more importantly time consuming. Frustrating the efforts of high speed processors with pendantic over administration is like having high performance sports cars which spend most of their time waiting at traffic signals, with only occasional bursts of frenzied activity. Consequently the interrupt handler was slimmed down, yet retained much of the information of the earlier system, was easier to use and much quicker. Central to its operation is a communications register (COMREG). Each physical unit whether it be a slave or VOS etc. has a bit position within the register corresponding to its BIO level. At start of day the register is cleared, but whenever a task is initiated the corresponding bit is set and remains so until it is serviced to remove its interupt. The bit positions for the VOS and the video scanner are fixed and known, but as logical slaves are used their bit location is not immediately obvious.

To set a bit in COMREG, there are several possible methods, involving the following routines.

SETCOM

Inputs ACC = LSN

Works out actual bit position from resource information and sets it.

VEC320

Inputs ACC = LSN

P2 Interrupt vector number

As setcom but also performs a slave hard interrupt

SETBIT

Inputs ACC = LUN

Sets associated bit position

When testing a status bit a mask must be derived to extract it from COMREG. For fixed positions this is known by the user but for logical slaves the following routines must be used

GETBIT

Input ACC = LSN

Output ACC = Bit Mask

Another routine of use is,

CLRBIT uses BIO level to CLR bit positions

INPUT P2 = BIO LEVEL

The main handler that tests and selectively deactivates task statuses is called PRIBIO. On entry the bit mask must be held in the accumulator and PRIBIO will not exit until the specified tasks have been terminated and their BIOs cleared by server routines. The procedure for PRIBIO is as follows:

CALL PRIBIO

1) AND mask with COMREG
2) If ZERO then go to 19 else go to 3
3) Call UCALD
    4) If BIO Active then 5 else return
    5) Call WHOSIT
        6) Read Priority: disable new BIO's, put LUN in PO: put BIO level in P2
        7) RETURN
    8) Open stream to Unit
    9) Call CLRBIT
        10) Marks test as done by clearing bit in COMREG, using BIO level given by WHOSIT
        11) RETURN
    12) Uses identified BIO level to address BI SERV which is a table for start addresses for board service routines

13) CALL SERVER

14) Device specific function carried out on unit, the minimum required is to clear the BIO line

15) RETURNS

16) Re-enable BIOs

17) RETURN

18) Branch to 1

19) RETURN

To prevent lock ups priority level 7 may be used as a manual abort or alternatively some timeout facility may be implemented on level 6. The PRIBIO enables servicing of specific tasks and logging such events, even when more significant interrupts are present. Even so, a good program strategy would ideally always perform tests on the most significant active BIO present to avoid unnecessary interrupt stripping.

Whilst the journalling qualities of COMREG may be increased perhaps substantially, the speed penalties would seem to discourage it. Of course the service routines may vary and a special case of which could be used for interslave communications mentioned in Chapter 2. The mechanism by which this might be performed would require each slave to have two buffer blocks, one each for receive and transmit. Each communications packet transferred would comprise source destination addresses and several words of data. LSNs would be used for addressing with NULL being an easily tested special case meaning empty packet.

Each slave would use variables TXCNT and RXCNT to maintain their buffers along with a flag (FLGCOM). FLGCOM would indicate to a master service routine whether the BIO generated, signaled an end of task or a communications request.

Slave and master procedures suitable for implementing simple communications channels are given in Appendix (J).

The method of communications mentioned can provide a link which as far as slave software is concerned, is transparant but at the expense of a time penalty. The time cost of such a transfer is significant because of the intermittent processor activity associated with the actual data transfer especially when buffers become full. It is in fact considerably quicker and easier to split long tasks with run-time communication requirements to smaller tasks with result passing fitted in between. This was the method employed in most of the application programs and has a number of additional advantages. Not only does it remove the risk of stack corruption which is a sensitive area of the TMS32010, but it also eases context save overheads and is easier to debug due to the reduced program module size. It may well be that a future application will demand the send/receive buffer communications, which could be investigated using the strategy proposed in Appendix (J). However a program that is arranged thus is unlikely to be satisfied by the performance of the software channel. To

date, a hardware mechanism has been impractical due to space limitations on the processor boards, but chip sets are now becoming available that implement simple LAN protocols very compactly indeed. If sufficient incentive was mustered then these could be interfaced on the slave peripheral I/O ports using the expansion connector. If a unique list of source and destination addresses were provided a slave could not only communicate with its nearest neighbour on a specific level, but any processor or station regardless of tree position.

4.10 PROGRAM FORMAT

To ensure that application programs are developed in a consistent and readable fashion, a set of de facto conventions have been adopted for layout. Some of these have arrived from subject preference and convenience although a number result from the limitations of the TMS32010 EVM and its line by line assembler. The leading section of program source will contain the label definitions, associating meaningful names with internal datamemory locations, primary ports and constants. The first program module to run is COEF which is a standard utility to load datamemory constants with values from a table in program memory, just prior to the routine itself. It would be inadvisable to start this table or any program for that matter, before location 8. This precaution is due to a design fault in the TMS32010 which makes it impossible to differentiate between TBLWS to locations 0 to 7 and

OUTS on the primary ports! After COEF, program flow branches much further on in address space to MAIN which is the major initialisation routine. Immediately following COEF is the table space, which is a section of program memory reserved for constant storage or as work space. This is defined early on in address space as the line by line assembler requires that labels used as constants must be defined before use, so tables must appear in program memory prior to the routines which reference them by name. After the tables comes SYBOOT and all the slave routines to be booted over. Between the endmarker SYFIN and MAIN are located all of the required utility subroutines. The purpose of MAIN is to initialise all system hardware and compute any special coefficients. Examples of the latter are the factors used to convert between different sets of screen co-ordinates, i.e. from the coarse GRID of the nomination rig to absolute co-ordinates or vice versa. If RSCE is actuated then MAIN is likely to be followed by several strategies for different numbers of slaves to perform similar tasks using alternative sequences. The actual tasks themselves are just collections of calls to the utility routines. A typical source layout would not be too disimilar to that shown below.

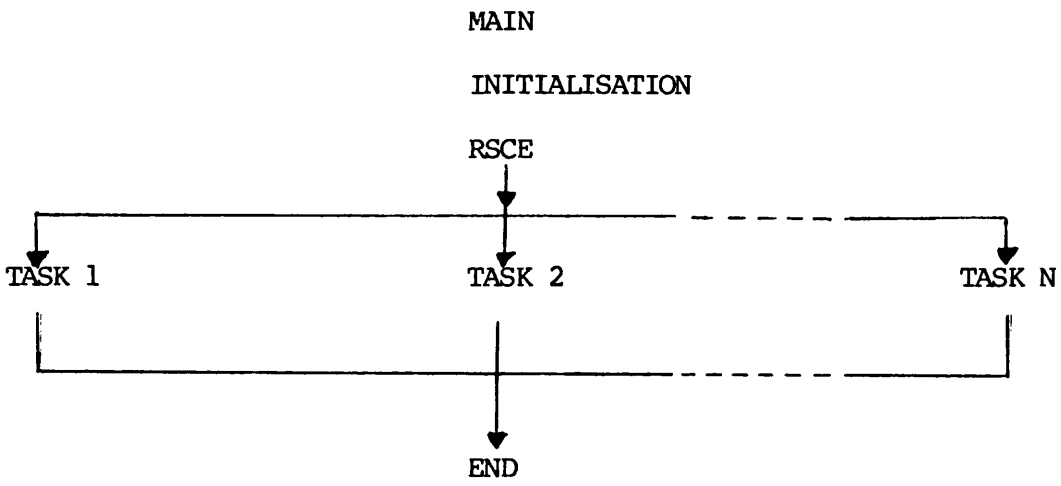
```
                    LABEL DEFINITIONS
                        AORG  8
                        COEF TABLE
START                   COEF
                        BRANCH TO MAIN
```

SYBOOT

SLAVE ROUTINES

SYFIN

SYSTEM UTILITIES



4.11 UTILITIES

A large collection of utilities are available but due to the small program space (4K x 16) only small subsets of the main library are installed for particular applications. Within this collection of routines there are two clearly identifiable sub-libraries, FLTLIB (floating point library) and DISPLIB (display library). It is perhaps unusual to find a floating point capability on a TMS32010, fixed point fractions being much more efficient.

The problem with fixed point is although it can perform admirably for most applications with skillful programming, it has problems when coping with variables of very large dynamic range. A

software implementation of floating point overcomes this, but its use is only justified if very rare, due to the obvious time penalties. The library provides the following capabilities.

FLTNRM Normalises fixed point to floating point number

FLTMPY Floating point multiply

FLTDIV Floating point divide

ALIGN Equates exponents of numbers, usually prior to ADD or SUBTRACT

FLTADD Floating point ADD

FLTSUB Floating point SUB

FLTINV Floating point INVERSE

FLTSQR Floating point SQUARE ROOT

(The last function uses the Lyusternick approximation followed by 2 iterations giving a specified accuracy of 28.99 bits see Ref. 46).

The DISPLIB library allows the user to generate artificial displays such as graphs and some limited text on the monitor screen. Strictly speaking this is outside the scope of VOSTTAC1 and should really be performed via the auxilliary display or downloading to a host. In practise it is often quite desirable to generate results displays within VOSTTAC1 especially when commissioning new programs. DISPLIB currently works on a 64x64 window using the following functions

CLRWIN Blanks window

CORDSC Scales waveform for graphical display

DRAXIS Draws X&Y axis in window

DWIN Modifies window to draw graph

GRPOUT Dumps modified window to specified screen position with X&Y zoom feature

ALPHA Modifies a string to represent text (with UDGs within 8x8 block)

PRINT Writes strings to window

Due to memory limitations the character set is very small and therefore only intended for program development rather than professional display.

The software described in this Chapter is only a vehicle to achieve some goal. Designed as a research tool for real time techniques, it was important that the program utilities provided, enabled VOSTTAC1 to confront image processing problems. A common requirement for high speed applications is to segment an image into interesting and uninteresting regions. Usually the entry point to this problem requires some form of thresholding and the next chapter is devoted to just that.

CHAPTER 5

THRESHOLDING AND REGIONS

5.1. THRESHOLDING

Early picture processing by computer is usually motivated by one of two objectives, either to locate a sub image of interest prior to further processing or to enhance the visual display presented to the human analyst. To take examples, the first might be required by some object detection or tracking system, whereas the second may be a necessity for perhaps studying satellite or biochemical images. Both goals would almost invariably require some form of thresholding. This is a convenient technique for segmenting images into disjoint regions, usually based on some pixel property criteria, such as grey level. The specifications for regions will vary in terms of accuracy and definition, depending on the application and the algorithm employed.

5.2. BINARY THRESHOLDS WITH COLOUR

The simplest and most common region segmentation method is the binary threshold. Basically all pixels with grey level below the specificed level are assigned to one group and those above to a second group. If these regions are assigned very different luminance levels, such as peak white and black, the classification produces a simple contrast enhancement. When the primary concern is to present an acceptable visual display the threshold level is subjectively and interactively chosen by the user. If however the segmentation is a preprocess for further computer analysis then the level must be more formalised.

Within controlled environments, as found in laboratory and factory inspection systems, thresholding is often all that is required for scene segmentation, due to contrasting objects and background under constant lighting conditions. If for example a white object is shown on a black background, splitting pixel luminance levels into equal to zero (=0) and greater than zero is sufficient. For multiple regioned images, steps of thresholds could be employed i.e. different grey levels to imply different objects. The use of colour is still quite rare, perhaps historically due to the cost of such systems. Fortunately this restriction is now easing due to technology advance, especially in the field of solid state colour imaging devices. VOSTTAC1 is a true colour system that deals with the three primary image spaces R, G and B, and should not be confused with equipment that works exclusively on luminance and then displays results using false colours. Obviously detecting the presence of a primary colour is identical to extracting the white on black object using a greater than zero test but applied to the relevant space. Similarly the secondary colours, i.e. cyan, magenta and yellow, can be detected from the combined results from such thresholds. Using the stores in such a binary fashion only results in a maximum possible 8 colours including black and white. For more complex colour combinations, more selective analysis of the R, G and B components is required.

In fact VOSTTAC1 can differentiate between a maximum of 4096 colours, although in practice smaller subsets would almost certainly be employed to make segmentation more robust.

5.3. HISTOGRAMMING AND ITS PROPERTIES

If the world was full of uniformly illuminated contrasting scenes, image processing would be fairly trivial. Predictably this is not the case and innumerable complexities, both natural and man-made always seem to conspire to make the task as difficult as possible. Lighting is a prime culprit when it comes to meddling with thresholding techniques. For example, a greater than zero test becomes worthless if the so called black level background is raised to grey by increased illumination and the effect on multiple grey level "steps" can only be described as chaotic. The apriori thresholds resulting from educated guesses would therefore seem inadequate and so some image measurement is required to assist in the determination of suitable levels. A well known approach is via the computation of histograms (refs 25,47,48,49). These are represented by waveforms which show the distribution of pixels into the possible grey/colour levels. They are computed by first assigning a "bucket" to each possible grey level and then every pixel within the sub image of interest is level tested and the relevant bucket contents incremented. The complete histogram is then usually a graph with grey level as abscissa and pixel count as the ordinate (although higher dimensional forms are used). Mathematically the histogram is best considered by an example. Consider a continuous 2-D image

function f(x,y) FIG (28a) which peaks at the centre of the screen and decays smoothly away at the boundary such as a gaussian, see FIG (28b). If some slice is taken through it normal to the peak at grey level G1, all pixels equal to or above this value will be shown and represent an enclosed area A1. If a higher level slice is performed at G2, A2 will be produced, within A1. This thresholded area function may be generalised to A(G). The histogram can then be represented as:

$$H(G) = \lim_{\Delta G \to 0} \frac{A(G) - A(G + \Delta G)}{\Delta G} = \frac{-dA(G)}{dG} \qquad (19)$$

Therefore the histogram of a continuous image is the negative of its area function.

By integrating both sides,

$$\int_G^{\infty} H(R)dR = -[A(R)]_G^{\infty} = A[G] \qquad R = \text{DUMMY VARIABLE} \qquad (20)$$

The integral of the histogram function from threshold level to max range is equivalent to the area function. The special case is when G = 0 and A(G) = area of the image in pixels.

If statistical image analysis is to follow the histogram computation, there are a number of interesting measurements available.

PDF probablity density function found by normalising the histogram with respect to area

CDF cumulative distribution function which is the area function normalised with respect to image area

IOD integrated optical density - measure of image "mass" given by:

$$IOD = \int_{o}^{\ell}\int_{o}^{m} G(x,y)dxdy \qquad l,m = \text{region delimiters} \qquad (21)$$

$$\text{DIGITALLY IOD} = \sum_{i=1}^{NLINES} \sum_{j=1}^{NCOLUMNS} G(i,j) \qquad (22)$$

G(i,j) is grey level of pixel at specific position

LET N_m = number pixels in image with grey level = m

then

$$IOD = \sum_{m=o}^{p} mN_m \qquad (23)$$

P = PEAK GREY LEVEL

Nm is equivalent to histogram calculated at level m

Therefore

$$IOD = \sum_{m=o}^{p} mH(m) \qquad (24)$$

MGL Mean interior grey level given by:

$$MGL = \frac{IOD(m)}{A(m)} \qquad (25)$$

M = THRESHOLD LEVEL

A() = AREA FUNCTION

Another useful property is that if an image contains several disjoint sub areas of known histograms then the whole image histogram is just the sum of their sum.

The example shown in FIG (29) is in fact a special case known as a BIMODAL distribution and is derived from an image containing two contrasting regions. Under such circumstances it is relatively easy to compute the optimum threshold merely by placing it at the bottom of the valley between the two peaks. Unfortunately even in controlled circumstances, images are not necessarily BIMODAL and real world examples are rarely so obliging.

5.4. POINT OPERATIONS

Histograms need not be fixed or constant as their characteristics may be modified by point operations on the image. A point operator is one which maps an input image to an output image so that each resultant pixel level only depends on the level of the corresponding input pixel. Sliding is a typical technique which moves the whole histogram left or right and is achieved by subtracting a constant from, or adding to the image pixels. Its visual effect is similar to that of the brightness control on a monitor or TV, either lightening or darkening the picture as required. No extra contrast is gained and can quite conceivably be degraded if too large an offset drives the levels into the peak white or black clippers. Contrast enhancement can be achieved by a point operator which stretches the histogram. This is a little more complex than the slide, requiring multiplications which raises a real danger of overflow especially if the multiplication logic can only work for powers of 2. Shrinking the waveform is also possible and achieved by division, although being harder to perform (with any accuracy) than the other operators it is less widely used.

Sliding and stretching techniques are often combined to produce general contrast enhancements, which forces images to make full use of the available dynamic range of sample levels. As both these processes require only simple arithmetic operators, they can be performed simultaneously. PICS (17,18) show a contrast enhancement using add and multiply. Notice that the text on the black file becomes visible in the modified picture.

Sometimes it is desirable to reflect the histogram by complementing the pixel levels so that black becomes white etc. Although this gives an unnatural appearance to the picture, similar to a photographic negative, it can make parts of the image more visible. The reason being is that the eye responds logarithmically to picture brightness so that fine details near peak white may normally be invisible prior to complementation.

The point operators discussed are typically in the utility arsenal of any image enhancer, LANDSAT (ref.50) processing being a prime example. However they are not obviously capable of forcing BIMODAL histograms from images which clearly are not and in any case such a strategy is unlikely to be justified in terms of information lost.

5.5. AREAS OF INTEREST

VOSTTAC1 supports many of the methods used for image enhancement, but it was never really intended for that purpose. Its primary role being to make some decision about an image rather than transform it for human analysis. As a consequence, observed scenes are assumed to contain at least one area of interest with noticeable features, which if not unique are rare amongst the regions of no interest. This is complicated as VOSTTAC1 is not confined to artificial environments and therefore is almost certainly dealing with multimodal pictures of varying confusion.

Suitable low level extraction features, might be the grey or colour levels mentioned or perhaps edge discontinuities and movement discussed in later chapters.

If colour on grey level is adopted then before the pixels of an image can be assigned to particular classes of varying interest, some model is required to characterise the classes themselves. Furthermore the overall goal of the segmentation scheme must be defined. For the purposes of some object detection, the initial goal is to identify an object region and compute its approximate screen position. The thresholding scheme to achieve this is not immediately obvious, as the assumed fixed level and bimodal technqiues used in controlled environments are inapplicable.

5.6. ON LINE ADAPTIVE MULTI-SPECTRAL THRESHOLDING

A facility exists within VOSTTAC1 to perform on line learning. In the context of thresholding it has the advantage of computing histograms and the eventual thresholds under the working conditions of illumination. It would then be possible to compute the global histogram, but should the object or region of interest be small with respect to the whole picture, which is not improbable, then it will have negligible effect on the PDF. Even if this were not the case and the image segmented perfectly, without some further knowledge, the computer has no way of assessing which of the regions is interesting. These problems may be overcome by using VOSTTAC1 in target nominate mode, wherein the first stage is to enclose the desired object area within a

boundary window shown highlighted on the auxilliary display. This boundary is usually a small rectangular box, but it is within the scope of the machine to trace an accurate perimeter around the image and thus improve the signal to noise ratio of the target by eliminating background effects. The chosen boundary shape has no effect on the histogram, which is spatially ignorant, but its enclosed area must be chosen so that it is dominated by the target, in which case the major peaks of the PDF will correspond to target attributes alone. If bandpasses are then constructed about the major peak or peaks for a monochrome image, then only those pixels within the grey level bands will be considered as members of the desired class. In real images this is only partially successful due to multiple regions of similar grey level. Results with multispectral systems are more encouraging. For colour visual systems extra effort is required as there are at least 3 colour spaces R, G and B for which histograms must be computed, all of which must then be peak detected in the same way as the luminance signal. Colour bandpasses are then positioned about the peaks providing a valid combination only if a pixel vector components for R, G and B all fall within these acceptable bounds. Sometimes an object is characterised by more than one colour combination which can be identified by repeated application of the peak finding and bandpass strategy. The technique yields several regions which in this instance typify the target and is sometimes known as a recursive splitting, region growing (ref 26,51). Using a tolerance of pixel values rather

than fixed levels made the classification less sensitive to quantization effects and to a limited degree, illuminance. The allowable departure from a peak was a compromise between avoiding these minor effects without seriously degrading the selectivity of the test. (For a 4-bit system the best ranges were 3 to 5 with appropriate correction near the clippers).

Significant lighting variations will violate the conditions for the multispectral thresholding mentioned above, therefore some compensating action is needed. The invariance of colour ratios is not a suitable property to analyse say a video sequence of images and would be more aptly named the "variance" of ratios. This lack of precision is due partly to non-linearities in the video system and to some extent multiple light sources of differing spectral range and spatial position. Another scheme is therefore required whch can retain the selectivity of the segmentation yet is tolerant to slowly changing environmental factors. The engineering solution is to adaptively modify the bandpass with time, on the assumption that if a feature is known for one frame, it will be valid for the next, although it may approach the extremetities of the class limit. By repeating the learning procedure at every processing period of 1 to N frames (where N is small ≤ 3) the classifier may be adapted to the prevailing localised ambient conditions most pertinent to the next segmentation attempt.

Apparent illumination changes are not caused exclusively by changes in ambient lighting. Observed objects may suffer varying degrees of shadow depending on their position within a scene, which becomes a variable for moving targets. This shadow may be naturally prevelant or artificially introduced by sources with non ideal flat field response. An example of the latter is the edge of screen darkening, due to camera aperture effects under conditions of low light. The aforementioned complications would seem to recommend local as opposed to global threshold techniques (ref. 26) and hence reinforce the use of target nomination and subsequent adaption. Referring back to enhancement, there is a common problem especially in biomedical applications of segmenting objects (perhaps chromosones) from slowly varying backgrounds and this too can be solved using local analysis.

5.7. LOCAL HISTOGRAMMING/THRESHOLDING

The approach arbitrarily divides the picture or area of interest into a grid of small squares. Histograms are computed for each sub area and some criteria chosen for level selection perhaps via the BIMODAL assumption. The thresholded regions may then be merged subject to a homogenity constraint. Alternatively a splitting and merging technique can be adopted (ref.52) which can increase but also decrease region size.

VOSTTAC1 is quite capable of windowing small sub regions as the preliminary step towards spatially variant local thresholding and subsequent region growing. Moreover the variable boundary

capability inherent in the VOS machine could support the download of local areas of irregular shape if demanded by the user or some earlier process. In addition the resolution reduction effect of VOSTTAC1 can be employed within a more hierarchical approach which involves region growing at coarse resolution followed by successive finer resolution passes to more closely examine boundaries.

5.8. SPEED FACTORS

The foregoing discussions have hopefully provided some insight into segmentation techniques via thresholding and proposed approaches where VOSTTAC1 can be used to advantage, namely multispectral recursive splitting and spatially variant or hierarchical thresholding. Whichever route is taken to arrive at a classification level it eventually must be applied to the data using some scheme subject to the VOSTTAC1 motto "If it can't work quickly change it!". Perusing textbooks on image analysis generally yields few guidelines about efficient implementation, mainly because off-line activities are the norm and thresholding responsible for a few seconds processing within a run time of hours is unlikely to merit much attention. However the application areas for VOSTTAC1 demand processing periods of a few frames, giving a total computing time of tens of milliseconds. Understandably every effort must be made to prevent any form of time wasting. To get a feel for the magnitude of the problem it is worthwhile considering some simple examples which highlight the merits of special hardware thresholding units.

At its simplest level thresholding may be accomplished by ANDing a designated mask with pixel values, so no matter what other processes are performed at least one extra instruction per pixel is introduced. This may seem a small penalty to pay but it is the instructions by pixel product that is important and for a full screen would require:

$$\begin{aligned} \text{NLINESxNROWS} &= 577 \times 360 \\ &= 207{,}720 \text{ INSTRUCTIONS} \end{aligned} \qquad (26)$$

A 5MIP slave would take approximately 40ms or one extra frame period to achieve this. Things aren't usually so bad, as smaller areas are normally considered and tasks split between multiple slaves. However the time is representative of a fraction of the overhead to produce on-line displays of image sequences, which is one of the desired properties for VOSTTAC1. The situation degrades dramatically if multispectral thresholding is implemented. Performing double edged thresholding on three colour spaces via software, demands many more instructions than the simple approach and when multiple colour combinations are considered the speed may be reduced by a factor of 40! Such a degradation would ordinarily demand that the approach be discarded, which considering its inherent advantages would be a great loss. Other researchers faced with this kind of problem, implement hardware at the "front end" of the machine,

so that the data is conditioned as it enters the store. Whilst this replaces the large instruction overhead with a negligible delay which is admirable, it badly restricts the stored data to such an extent that many processing options are lost. Firstly its effect is global, which prevents the much desired, spatially variant techniques described earlier, and secondly all picture data is discarded apart from the resultant binary segmented image. The latter implies that no further processes can use the raw data including the generation of the next threshold, until a new frame of information is captured without threshold. Ideally what is required is a means of obtaining the speed advantages of the front end threshold without, the inherent data restriction. Fortunately the DTP in conjunction with the VOS machine can do just that.

5.9. THRESHOLDING AND MAPPING WITH THE DTP

Every DTP contains bandpass proms, comparators and a colour threshold prom configured to apply multispectral thresholding. From histogramming analysis a particular peak is identified and its level applied to the bandpass prom which then computes the upper and lower limits with clipping to implement the bandpass comparison. In fact, two duplicate sets of hardware are provided to accommodate double peak systems. The segmentation decision flag is then derived from an OR function on the outputs of the two circuits and fed to the colour threshold prom. If working in standalone mode this is all that is required to condition the dump

data as all three primary colours function independently. This method is acceptable for artificially controlled situations with very colour distinct objects. More general use requires a coordinated effort facilitated via a communications link between store cards, allowing "TRUE" output only if all three primary thresholds are satisfied, ie a specific colour combination. Although the DTP has so far been discussed relative to a multispectral technique, it is sufficiently flexible to implement any simple binary threshold.

Because of its position within the system the DTP only effects data that is being dumped by the VOS and therefore overcomes both problems of front end processing. Spatially variant thresholding is supported due to the very flexible windowing capabilities of the VOS and because data is only read from a store, the original contents remain unchanged.

The goal of thresholding is to identify separate regions, usually by transforming to a binary image which can then be conveniently examined to find active pixels via set/reset tests. This leads to a conflict of interests as thresholding should really only be used to flag those pixels which belong to a specific group, not necessarily to replace them with the flag itself. Discarding the finer detail is an obvious inefficiency especially when from a knowledge of the segmented region a slave might be required to perform some process on the original data therein.

Alternatively it is quite possible to gate the picture information so that if it satisfied the threshold it would be allowed through unmodified to the slave window as opposed to merely sending a binary value. The problem then becomes how to tag the picture areas that failed the test, as otherwise any pixel in the output window could be picture data. Level 0 (black) was reserved for pixels of no interest, but to prevent confusion with actual black pixels, these were transformed to level 1. This slight "liberty" is justified when considering the advantages gained. Not only is the data allowed through to the slaves, without noticeable logical or visual degradation, but also the binary image is present because the greater than zero test is still valid. Coupled with the fact that thresholds are applied without microprocessor involvement and only introduce a negligible delay (order of ns) as opposed to many multiseconds lost, this section of the DTP is a very great asset to VOSTTAC1.

Although microprocessors are not required for data thresholding they are needed to compile the histograms. Obviously to do this for a full screen at maximum resolution would, by VOSTTAC1 standards, be slow (order of seconds). Therefore the process is normally applied over a restricted area window nominated either by the user or as a result of some previous computer action. The use of such reduced areas allows histograms to be computed every frame if required to facilitate adaption without introducing unreasonable overheads. Larger areas can be easily involved in

the histogramming exercise providing reduced resolution is acceptable. The alternative is to accumulate the histogram over a number of frames so that the action does not overburden critical loop processing.

5.10 BLOB SEGMENTING

It would be naive to expect that even multispectral thresholding would only ever show points within the desired object boundary and so some further process is required. If the rogue pixel areas are small with respect to the target, which is often the case after selective colour thresholding they may be eliminated by successive low pass or averaging filters. Even if some residual noise remains, the average centre of the region will still approximate to the valid target centroid and it is this rough positional information that is of interest. The problem becomes more awkward, when there are several objects present all of similar size. Filtering is then of little use and a further level of segmentation at "blob" rather than pixel level is needed. Blobs (ref. 53,54,55,56) being subregions whose important attributes are usually central coordinate (screen position) and area, but not normally shape. In the context of object detection or tracking, these initial measurements need not be exact, but provide a rough guide as to where to perform further examinations. There are many ways of tackling this, some involving boundary measurements as an

intermediate step. Boundary finding using differential edge operators is discussed in Chapter 6. A simple technique which does not really fit into this category, but worth mentioning is Paperts Turtle (1973) which proceeds as follows:

1 Scan image until pixel is encountered

2 If it is a region pixel, turn left and step else turn right and step

3 Terminate on reaching the start point

Under ideal noise free conditions of 4 connected regions this would provide a boundary map from which area and centre could be calculated. Unfortunately the process is very sensitive to noise and sometimes disastrously so and usually requires operator supervision. This fault makes it inapplicable for many situations, but should suitable conditions prevail, the simple jumping and stepping nature of the test would allow relatively straightforward hardware implementation.

What is really required for blob segmentations is some sort of averaging process, working on blob area which will be less noise sensitive than simple boundary specifications even at the expense of some positional precision. Region growing can do just that, usually by means of some splitting, merging or combined technique. The choice of strategy is constrained largely by the speed demands of VOSTTAC1. More precisely the algorithm must segment the 2-D space, using 1-D scans (of variable orientation)

and partial results. Furthermore any derived lists should be searchable in a sequential manner. In this way the slaves could work with optimum efficiency but less obviously, the processing environment would be ideal for high speed hardware implementation. A procedure which satisfies these criteria is shown below, and relates to FIG (30) (derived from PIC (31) prior to edge detection).

1) SELECT RESOLUTION - USUALLY COARSE GRID OF 64 X 64
2) NOISE FILTER DATA (OPTIONAL)
3) DIVIDE WINDOWED AREA INTO SUB BLOCKS. EACH 4 X 4.
4) FOR EVERY ROW OF EACH SUBLOCK AN AREA IS COMPUTED AND ALSO LEFT CONNECTIVITY AND RIGHT CONNECTIVITY FLAGS, FORMING A SEQUENTIAL LIST (FOR 64X64 WINDOW LIST HAS 1024 ENTRIES)
5) LIST IS SCAN SEQUENTIALLY, COMBINING THE 4 ROWS OF THE SUBLOCKS TO FORM GROUPS, WITH ATTRIBUTES OF AREA LEFT, RIGHT, TOP AND BOTTOM CONNECTIVITY FLAGS (256 ENTRIES)
6) SCAN THE LIST SEQUENTIALLY (ALTHOUGH EQUIVALENT TO VERTICAL SCAN). SEARCH FOR NEW GROUPS, WHEN FOUND ASSIGN AND STORE A UNIQUE CODE NUMBER, IF THE NEXT NEIGHBOUR IS CONNECTED THEN IS GIVEN THE SAME CODE, ELSE CODE COUNTER IS INCREMENTED AND SCAN MOVES ON. (CODE 0 BEING SPECIAL CASE FOR NULL BLOCK). THIS STAGE GENERATES VARIABLE LIST OF 256 ENTRIES.

7) CODE LIST SCANNED (EQUIVALENT TO HORIZONTAL) USING BOTTOM AND TOP, CONNECTIVITY TESTS. IF A HIGHER GROUP CONNECTS TO A LOWER GROUPWITH A DIFFERENT CODE, THEN THE BOTTOM SET IS ABSORBED INTO THE TOP BY OVERWRITING ALL OCCURRENCES OF ITS CODE.

8) TO FIND CENTRES OF REGIONS, SEARCH FOR ALL GROUP MEMBERS, COMPUTE TOTAL AREA, PRODUCE X&Y COORDINATE ARRAYS AND PERFORM INCREMENTAL SEARCHES IN X&Y DIRECTIONS UNTIL 1/2 AREA LINES ARE FOUND, AND HENCE THE CENTRAL COORDINATES.

Because the procedure is so regular it is suitable for hardware construction. For example consider stages 4 and 5. A four stage notional window could be passed along the sub area rows and for each sub block position the area and connectivity flags could be generated and added to a list. This could be achieved by a small clearable delay line and perhaps a prom. Row accumulation could also be by prom requiring 4 lots of 3 bit area measurements necessitating a 12 bit address or 4K prom. The advantage of using proms as opposed to discrete logic is that the central area and connectivity flag generation may be quite complex without introducing significant time penalties. VOSTTAC1 currently uses a software approach which whilst understandably slower than the hardware solution is sufficiently swift to be useable. The allocation of processing time as a single slave activity is put forward below.

(Assuming Connectivity via Adjacency):

Stage 1

1) Negligible Penalty
2) 1 to 3 mS - this high speed is only possible because an averaging function is an inherent DTP process (See Chapter 6) 1 mS
3) Negligible Penalty
4) For 64x64 Window requires 6 mS
5) For 1024 entry list requires 2 mS
6) For 256 entry list (42 active blocks) requires 1 mS
7) Difficult to define generally but for the example with 6 targets 6.8mS
8) Again difficult to define generally but for the example 3.4mS

Overall Total 15 mS

N.B.: Segmenter Timing discussed in detail in Chapter 8.

The search is basically a block averager followed by a 4 connectivity merging and eventual blob centering. Many modifications are possible, perhaps the inclusion of 8 connected merging, although this is not obviously hardware compatible. In theory the technique could be applied without the averaging, direct to the raw data. One of the problems with this is that 8

connected objects would have less chance of being observed, without being summed locally. Perhaps more importantly, more time consuming searching activities would be required prior to any data reduction which would significantly degrade performance.

The solution to accessing 8 connected regions in an easy methodical way is by no means straightforward, however by using vector scanning it is possible to produce some tests along directions other than horizontal and vertical. By scanning at 45^{o} or -45^{o} the diagonals become the means of connection, i.e. if a 4 connected merge is applied to such angled data it is actually merging along the diagonals. This may be useful property when images are highly diagonal, i.e. parallel 45^{o} lines.

Within this chapter a possible scheme of target segmentations has been proposed via multispectral thresholding derived from a recursive splitting technique and followed by a hardware suitable blob merge and split. There is an alternative or dual approach which analyses local edge discontinuities as a means of early segmentation and is discussed in the following chapter.

CHAPTER 6

EDGE DETECTION AND FILTERING

6.1. EDGES

In general, problems may usually be tackled by following one of several approaches and image segmentation is no exception. Two philosophies tend to dominate, either region or edge based. This discussion is concerned with the latter in order to produce some comparison with the multispectral split and merge region techniques mentioned in the previous chapter.

Boundary or edge detection algorithms detect discontinuities in fixed level between disjoint regions. If cases are restricted to luminance only, then edges would be observed as marked changes in grey level. To make these more dominant and noticeable it is common practice to transform the image into an "edge" picture by attenuating the low frequency components and enhancing the higher ones. This seemingly innocuous procedure is sufficient justification for reams of technical papers proposing, comprising and applying different techniques (surveyed in refs 57,58,59). Generally these may be put into four classes.

1 PARAMETRIC MODELS

2 HIGH LEVEL CONTEXUAL EDGE DECISIONS

3 LOCAL LINEAR DIFFERENTIAL OPERATOR TEMPLATES

4 LOCAL NON LINEAR RANGE FILTER TEMPLATES

How is an edge defined?

Parametric modelling, of which the Heuckel (ref.60,61) method is the most well known, attempts to answer this by comparing an actual edge with a set of models or basis functions shown in FIG (31) subject to an error criteria such as the least mean square. Minimising the error is no simple task as several variables are involved and the technique tends to be complex. Referring again to FIG (31) it can be seen that the functions are circular which makes practical implementation very difficult. Although large fixed grids can approximate to circles and efforts have been made to reduce the complexity of Heuckel's method (ref.62) the technique remains too awkward, slow and unwieldy for high speed real time application. So too is the next method on the list. High level global decision making about complex scenes at high speed is difficult enough, but coupled with the lack of apriori information on which to base judgements, successful application to real time operation would be generally improbable. If it were practical, such tests could help find edges that do not exist at local level or perhaps at any level! The last remark is not as nonsensical as it sounds, and humans quite often see lines which cannot be measured as grey level discontinuity. A phenomena known as lateral inhibition causes the neighbours of an excited retina region to produce a negative contribution similar to the response shown in FIG (32). Physically this can manifest itself as light

points/lines in dark regions or dark ones in light, at positions where no measurable discontinuity exists. A class of linear operators for which this is also true was proposed by Marr & Hildreth (ref.36).

According to this now famous work edges occur ar different scales and should therefore be tested as such. Central to their proposal was to filter the image by convolution with a Gaussian function to set up the right "scale" and then apply a second partial derivative operator in the form of a Laplacian function.

6.2. 2-D CONVOLUTION TEMPLATES

Their use of the Laplacian as an approximation to $d^2f/dx^2+d^2f/dy^2$ is perhaps why it is so well known as a small convolution template. The usual way in which these are applied is to pass the much smaller function template over the image and at each position compute the output pixel from the convolution sum of the pixels covered and weighted by the template coefficients as shown in FIG (33). The larger the template the more control is afforded but speed is degraded as the number of multiply adds per pixel is increased. In addition, too large templates can cause problems with screen boundary conditions either drastically reducing image size or giving invalid results in peripheral screen areas. For example using an MxM template on a NxN screen with no overlap produces a (N-MxN-M) output image. For small templates this is not too much of a problem, but if desired this restriction can be eased by "wrapping" the axes either directly or with a line drop.

VOSTTAC1 will wrap from say the right of the screen to the left of the screen with a single line drop, but with a bounded Y direction. The latter restriction is by no means severe as it is normal to have otherwise redundant screen lines to be used for this purpose. As far as high speed constraints are concerned practical 2-D templates are advisably limited to 3 x 3. A list of more common templates is given in Appendix (K) (which include functions other than edge enhancers). The transition from linear equation to approximate template is fairly straightforward but not often discussed in text books. Referring back to the Laplacian as an example the first step is to translate to discrete form and then implement the function as a spatial grid, ie

$$L(x,y) = \frac{d^2f}{dx^2} + \frac{d^2f}{dy^2} \qquad (27)$$

where

L = Laplacian

f = 2-D image function

For the discrete case

$$\frac{df}{dx_1} = f(x+1,y)-f(x,y) \qquad (28)$$

along the same line

$$\frac{df}{dx_2} = f(x,y)-f(x-1,y) \qquad (29)$$

Similarly for y

$$\frac{df}{dy_1} = f(x,y+1)-f(x,y) \tag{30}$$

along the same line (31)

$$\frac{df}{dy_2} = f(x,y)-f(x,y-1)$$

$$\frac{d^2f}{dx^2} = \frac{df}{dx_2} - \frac{df}{dx_1} \tag{32}$$

$$= [f(x,y)-f(x-1,y)]-[f(x+1,y)-f(x,y)] \tag{33}$$

$$= 2f(x,y)-[f(x-1,y)+f(x+1,y)] \tag{34}$$

$$\frac{d^2f}{dy^2} = \frac{df}{dy_2} - \frac{df}{dy_1} \tag{35}$$

$$= [f(x,y)-f(x,y-1)]-[f(x,y+1)-f(x,y)] \tag{36}$$

$$= 2f(x,y)-[f(x,y-1)+f(x,y+1)] \tag{37}$$

$$L(x,y) = \frac{d^2f}{dx^2} = \frac{d^2f}{dx^2} + \frac{d^2f}{dy^2} \tag{38}$$

$$= 4f(x,y)-[f(x-1,y)+f(x+1,y)+f(x,y-1)+f(x,y+1)] \tag{39}$$

If the coordinate (x,y) is regarded as the central coordinate of the template then this can be represented as:

f(x-1,y+1)	f(x,y+1)	f(x+1,y+1)	0	-1	0
f(x-1,y)	f(x,y)	f(x+1,y)	-1	4	-1
f(x-1,y-1)	f(x,y-1)	f(x+1 y-1)	0	-1	0

This gives a 4 connected approximation to the Laplacian. Similarly for an eight connected mask:

-1	-1	-1
-1	8	-1
-1	-1	-1

The Laplacian mask has not been applied generally with any great distinction. Moreover its critics claim that its lack of directional information and noise sensitivity make is unsuitable for use. The second point is to some extent unfair because it usually arises from applying the function to an image that has not been Gaussian bandlimited, a factor essential to the arguments of Marr & Hildreth.

The reason for this omission is not hard to find. Image filtering using the Gaussian is very time consuming and for the lower frequency operations, a template version would require 8 x 8

dimensions. Even ignoring the actual calculations, 64 memory accesses are required to compute one output pixel and for a 577x360 screen say, would require $64x577x360 = 13.3X10^6$ operations and considering the likely complexity of these operations a software instruction count would be much higher. So for systems where speed is a premium, edge enhancers are applied direct to the image or perhaps after some very elementary prefilter stage. An example of the Laplacian 8 connected mask applied to a complex test image is shown in PICS (19,20). Note that because of its directional insensitivity it produces boundary information in one pass. Despite its limitations, when suitably thresholded, the Laplacian tends to give a "cleanish" uncluttered display which if only partially suited for further computer analysis seems to make some sense to the human eye. Even if this function were the crème de la crème of differential operators, it still might not be chosen for the very high speed real time systems.

To clarify the last remark consider FIG (33) which shows a 3x3 template being convolved with an image. For every one output pixel 9 pixels have to be addressed followed by multiplications, additions thresholding, and clipping all performed before the O/P pixel is addressed and updated. Aside from the convolution calculations the memory pointer is having to jump frantically around in address space, so its calculation is a problem. General

software solutions are therefore quite slow, but so too are modest hardware solutions. Special purpose 2-D template convolvers have been proposed (ref.63) but as yet not produced.

6.3. SEPARABLE TEMPLATES

From the designer's point of view, an easily implemented template could be applied by regular addressing patterns, i.e. scan lines and require integer or truncated fraction arithmetic. Fortunately some templates can be broken down into partial products by performing scans parallel to one direction followed by a 2nd pass in an orthogonal direction. For a template to be easily separable in this way, it must have similar rows or colums (or related by simple multiplicative constants), and so the Laplacian is clearly not separable. From the list prime candidates include Kirsch/Prewitt, Sobel, Vlinseg & Hlinseg operators. Of these the first is the simplest and most common. (The mask itself is accredited to Prewitt in some literature and as a subset of Kirsch templates in others, so they are both mentioned to avoid offence). Examples of the Kirsch/Prewitt templates are shown in PICS (21,22) for which the output value is actually divided by 2. The mask tends to pick up many discontinuities which the user then has the option of thresholding to strip off the minor ones. Certainly differences of one grey level should be avoided as these could be produced by quantization effects. Unlike the Laplacian two templates are required, although they are much easier to perform. Phase 1 of a Kirsch/Prewitt template would convolve each

pixel along a line with the (-101), 1-D template.Phase 2 would then pass over these partial results in the orthogonal directions with say a 1/3, 1/3, 1/3 averager, to compute the actual pixel output. It is by no accident that the VOS machine can scan in this fashion at a number of orientations. This technique is generally suited to its image processing environment and requires all operations to be performed on sequential data streams - directly suited to the TMS32010 and hardware methods in general. The Sobel effects shown in PICS (23,24) are very similar to the Kirsch/Prewitt. (Sobel is generally accredited with this template although a paper reference appears elusive). In fact Abdou & Pratt (ref.64) concluded in their comparison of edge enhancement techniques that the two performed almost equally well as a function of edge orientation and signal to noise ratio. Practically the two are implemented in much the same way, the first pass being identical, but the orthogonal averaging proceeds with different integer constants. These operators are very common in high speed systems and the Sobel is currently used by British Aerospace as part of their military target tracking system developed over the past 7 years with very many man years of effort, (ref.65). Cussons (ref.66) whose work is relevant to the British Aerospace system concluded that the Sobel was better at locating low contrast blobs under his set of conditions. It is not uncommon to read several researchers claiming different operators as being "the best", but who is right? Usually they all are, it is the conditions of test that vary. What should be

gleaned from this is that the actual application will dictate their relative performance. For something fairly general purpose such as VOSTTAC1, there is little or no apriori image information prior to on-line learning and so optimum yet easily applied mask values are difficult if not impossible to define. Therefore adopting the Sobel and Kirsch/Prewitt templates as standard utilities would seem to be the most logical course of action. Can these be rivalled in terms of simplicity or performance by the other small template functions? The Roberts operator certainly seems simpler being only a 2x2 grid, but is it? Consider FIG (34) for which the gradient is given by:

$$M(x,y) = ((\Delta y)^2 + (\Delta x)^2)^{\frac{1}{2}} \qquad (40)$$

$$\text{PHASE } \theta(x,y) = \text{ATAN } (\Delta y / \Delta x) \qquad (41)$$

This can be achieved by passing the characteristic templates over the image, eg:

0	1	-1	0

and/or

-1	0	0	1

The Roberts is an approximation to a first derivative similar to the Kirsch/Prewitt & Sobel but not the Laplacian which is a second derivative operator producing zero crossings at edges. The Roberts can be written as:

$$R(i,j) = \nabla I(i,j) \tag{42}$$

which for the discrete case:

$$|R(i,j)| = \sqrt{(g(i+1,j+1)-g(i,j))^2+(g(i,j+1)-g(i+1,j))^2} \tag{43}$$

$$\text{ANGLE} = \frac{\pi}{4} + \text{TAN}^{-1}\left[\frac{g(i,j+1)-g(i+1,j)}{g(i+1,j+1)-g(i,j)}\right] \tag{44}$$

FOR THE GRID POSITIONS BELOW

i,j	i,j+1
i+1,j	i+1,j+1

This method is more susceptible to noise than the other two as the derivative is taken over a 1 pixel separation and there is no averaging of adjacent lines to compensate. The effects of the Roberts templates are shown in PICS (25,26). Note that the standard coefficients give a low "gain" output. Obviously increasing the values would counteract this but would also amplify the unaveraged noise. Because the technique works on the diagonal its effect is most noticeable on 45^{o} orientated objects, such as the central bar in the picture. Its limitations are characteristic of unaveraged 2x2 templates and another conclusion

of both Abdou & Pratt and Cussons was that they were inferior to the 3x3 versions. In terms of implementation, the standard template is not separable in the X&Y orientation as is the norm and cannot be applied in the same way as Kirsch/Prewitt & Sobel. Trying hard to find some inherent advantage yields a possible ease of use if specialised but limited hardware available as the technique only requires one subtraction, ie no multiplications or additions required due to the simple coefficients.

Gradient techniques allow more variable control of the orientation of test than those previously mentioned. The effects of a subset of operators can be seen in PICS (27-30). Compared with the Kirsch/Prewitt & Sobel functions the output is multiplied by two to be visually noticeable. This is because they have less gain due to testing over a smaller more specific region. As far as enhancing image edges they seem to have little to offer, however their strength lies in the inherent orientation of the results produced which can be a great asset to further segmentation and shape tracing. Unfortunately, they are not easy to apply in 2 passes. So what can be done if angular edge detection is required along directions other than the horizontal and vertical?

6.4. VECTOR SCANNING EDGE DETECTION

It was mentioned in Chapter [3] that the VOS machine is quite capable of producing say a 45^0 block transfer from picture to slave memory whilst maintaining a sequential data stream. As far as a slave processor is concerned, there is absolutely no

difference between such data and that derived from the horizontal or vertical directions. Therefore it is quite possible to apply a 3x3 template such as the ubiquitous Sobel to enhance edges along a diagonal. PICS (31,32) show images that have been enhanced in such a manner. The effects are characteristic of a superior Roberts technique, which now includes adjacent line averaging and extends the span of the test improving noise immunity. To do just this exactly using a square template and without vector scanning would necessitate a dimension of at least 5x5 whch would not be separable. All that must be remembered is that the template has larger spans than the usual operators which for 45^0 is increased by a factor of $\sqrt{2}$. PICS (33,34) show these scans applied to some snow tracks (frame from RARDE video). Note that the tracks are almost invisible to one scan but are more pronounced on the orthogonal version.Therefore this use of vector scanning is a means of producing variable angled edge detection all of which can be achieved by 2 pases of 1-D templates.

6.5. GENERAL CONVOLUTION MASKS

Whilst on the subject of linear templates it is worth considering some alternative convolution masks (see Appendix (K)). These include averager/LPF, HPF and line detectors. The effects of the line detectors are shown in PICS (35,36). Because of the small size of template they only really search for very thin bars, although they do produce edge enhancement effects they are not really suitable for general application to raw images. Using resolution skipping could possibly bring larger lines in to the

range of the small template, but usually line detectors are employed as a second stage of processing, i.e. grouping results of previous edge detections into lines.

Consider the high pass filtered PIC37. Because the template coefficient does not sum to zero, when placed on a uniform background the convolution sum will be equal to the covered grey level. This gives the filter an output which is quite acceptable to the viewer, enhancing edges yet retaining some low frequency information. If a human operator is required to monitor or assist in an edge detection, this may well be the best compromise, between original image representation and edge emphasis.

The LPF/Averager of PIC (38) is one of several possible variants. A common one uses coefficients of 1/9 which is fine, providing the unit that performs the intermediate mathematical function can support some form of division, not all can. The one used to produce PIC (38) employed mask values that allowed simple binary shifting to carry out the divisional functions (see Appendix (K)). Although this function helps to eliminate noise perhaps prior to an edge detection, it has a tendency to blur the picture. Seeing as how edge detectors respond to spatial high frequencies, this is obviously a degrading effect if extreme. There are however non-linear filters which preserve discontinuities yet attenuate spike noise, (refs 67,68,69,70,71). An interesting subset of this class is known as the rank filter.

6.6. NON LINEAR FILTERS AND EDGE DETECTORS

Up to now all the filters mentioned have been linear with their outputs derived from a convolution sum of pixels and coefficients. A class of non-linear operators are known as ranking filters. Essential to their operation is to take the pixels from a small window (similar to the linear templates) and sort them into ascending order by virtue of their luminance values. A filter of rank(N) gives as the output the value of the pixel at position N in the sorted list. Any rank number may be used but the most common are median, maximum and minimum. The median is the usual substitute for low pass filtering. An image has its "spike" noise reduced yet returns most of its edge information, see PIC (39). Note that repeated application of such operators tend to give the image a "painted on" appearance. The maximum and minimum pictures, PICS (40,41), are not strictly speaking genuine! Using the extreme values of rank tends to result in noise introduction, so values 1 place in are used. If PICS (40,41) are considered with respect to PIC (39), it can be seen that repeated application of the maximum filter grows bright yet shrinks dark regions and conversely the minimum filter has the reverse effect. This provides a means of emphasising discontinuities and there is considerable interest in a set of non-linear edge detectors which output the difference value between two rank positions, known as range filters. Recent work

by Hodgson & Bailey et al (refs 72,73) have presented the properties of rank & range filters in a very clear and concise manner, from which a subset of rank properties is given below.

1) Rank filters smooth noise

2) Rank filters charge mean intensity (except median)

3) Rank filters preserve edge shape

4) Rank filters shift edge position (except median)

5) Rank filters change all signals except a constant

6) Rank filters do not introduce new intensity values

Rank filters are often combined to achieve functions,

eg

LPF/noise suppressor = g = min (max (max (min(f)))) (45)

LPF/noise suppressor = y = $R_1(R_2(R_1(R_9(R_8(R_9(f))))))$

(46)

band pass g =

$$\min_n(\max_{2n}(\min(f)))-\min_k(\max_{2k}(\min_k(f)))$$

(47)

high pass $y = f-\min_n(\max_{2n}(\min_n(f)))$ (48)

A more comprehensive discussion is given in (ref. 72). In a further work (ref.73) range filters are compared with the now familiar Sobel, which reached the following four conclusions:

a) Range filters are more flexible than Sobel

b) Range filters contain no directional information only slope whereas Sobel provides both

c) Range filters perform better than Sobel when edge position is uncertain

d) Sobel better on severely corrupted images

The consensus of opinion on these non-linear operations is that they can provide very interesting processing capabilities, the possibilities of which have even now not been fully exhausted. Why then are they kept overshadowed by the much more common linear templates? The answer as far as high rate processing is concerned is down to speed. These non linear filters are slow, due almost entirely to the sort requirement. Even for a 3x3 window, the 9 value sort per pixel, degrades processing speed markedly compared with linear techniques. Using a bubble sort would require 64 comparisons per output pixel. Obviously faster sorts are available, but perhaps are not quite as regular and therefore not as hardware realisable as the bubble sort. Several proposals have been made to speed up the process, (ref. 70,74). These basically make use of the fact that as the window/template moves only one position (say 1 Δx) at a time per pixel, then for a 3x3 box only 3 pixels are lost and gained at each stage. Therefore a histograming technique may be used which just requires that new for old entries are changed using increments and decrements. While this undoubtedly gives a significant speed up factor as far as real-time frame rate analysis is concerned this makes the technique quite slow as opposed to extremely so. To really make

this method practical would require significant effort and investment to produce special VLSI to implement the sorts at high speed. Encouragingly such a program has been proposed (refs 72,73) if not yet begun.

Although linear techniques currently appear to be unrivalled in terms of speed it should be appreciated that if the ranking sort was finally achieved at high speed, a whole family of filters would be generated simultaneously as the masking and subtraction for the various output configurations is trivial.

Other edge detection schemes worth mentioning in passing are moment based, trimmed filters and gated filters. Moment based filters (refs. 66,75) have a statistical feel usually calculating some variation of a pixel value from the mean over a small window. Some central moments are shown below.

THIRD MOMENT

$$I_o = \frac{1}{N} \sum_{j=1}^{N} (X_j - \bar{X})^3 \qquad (49)$$

X_j = A PIXEL GREY LEVEL WITHIN WINDOW

$\bar{X}$ = AVERAGE GREY LEVEL

N = NUMBER OF PIXELS IN WINDOW

2nd

$$I_o = \frac{1}{N} \sum_{j=1}^{N} (X_j - \bar{X})^2 \qquad \text{(VARIANCE)} \qquad (50)$$

Trimmed and gated filters are just modifications of linear ones. Trimming merely removes pixel values which are very far removed from the median to reduce noise sensitivity. Gating is a means of selecting one of several linear strategies based on some local comparison of a pixel with its neighbours.

6.7. EDGE DETECTION SPEED REQUIREMENTS

Given that currently linear edge detectors are necessary for high speed operation and that separable versions are required for very fast implementation VOSTAC 1 must be performance tested under these conditions. Referring back to Chapter 2, a simple 3-stage 1-D convolution program was shown which required 16 cycles per pixel. For practical edge detection this value would be at least 20 cycles taking into account a slower outerloop when accessing a window. Therefore 20 cycles are required per pixel per pass, say.

TOTAL OPERATIONS FOR FULL SCREEN

= Nº LINES x LINESIZE x Nº CYCLES x Nº PASSES (51)

= 577 X 360 X 20 X 2

= 8.31×10^6 CYCLES (= INSTRUCTIONS)

If one frame time operation is required this must be achieved in 40mS.

.. INSTRUCTION RATE

$$= \frac{8.3 \times 10^6}{40 \times 10^{-3}} = 2.1 \times 10^8 \quad (52)$$

ie 200 MIPs

In addition there is the dump overhead which accounts for ∠20mS if edge detected windows are discarded after analysis or ∠40mS if loaded back to screen. So even with a 200 MIP processing capability at least 2 frames would be required. Bearing in mind that one TMS32010 instruction is often equivalent to several on general purpose machines, the enormity of the task can be appreciated. With a full complement of slaves the current version of VOSTTAC could only peak at 25MIPs (5 per processor), which although normally quite respectable is inadequate for producing edge detected images in 1-2 frames.

Obviously if a reduced area is considered, the time taken is much reduced and the software approach acceptable. However it is difficult to get a feel for moving sequences of images unless the edge detected output is continuously visual, an inconvenience normally tolerated by operators due to lack of choice. VOSTTAC1 has the option to generate such throughline displays by using the DTP.

6.8. EDGE DETECTION WITH THE DTP

The Data Transfer Processor contains amongst other things a template passer in the form of a 3 stage convolution filter. Normally the data out of a frame store cannot be guaranteed sequential along different angled scan rasters and therefore 1-D filtering techniques do not generally apply. However, as this has been an essential goal of the VOS machine, such streams are readily available. As the data leaves the store it passes through a 3-stage delay FIG (35). The delayed inputs are acted upon by the template prom, which for a given device holds four software selectable masks. The prom provides multiplication addition and output conditioning to achieve the desired 1-D convolutions and so the circuit is compact, all the effort being in the look up table generation. In this way the phenomenal MIP requirement is satisfied at the expense of a small delay of the order of ns. Note that the second pass is merely an orthogonally produced scan, using a different averaging template from the prom to implement perhaps the Kirsch/Prewitt or Sobel edge enhancers. Because of the capability of the VOS machine, the techniques are also

applicable to other angles including the much improved and modified diagonal techniques.

Speed wise the VOS/DTP combination only requires the dump overhead, and so by comparison with TMS32010 instructors the hardware peaks at 200MIPs! Compared with more general processors this parameter could easily reach the GMIP! value. This is achieved with no slave processing whatsoever (although they may continue to perform other procedures during the dump if required). Whilst on the theme of speed factors and sensationalism, a further comparison can be made. A project was set up to perform edge detection using a BBC model B computer, ref (76). To perform a comparative task , took approximately 6 minutes. Therefore the speed up factor relative to VOSTTAC1 is:

$$\frac{T(BBC)}{TVOSTTAC} = \frac{60 \times 60}{40 \times 10^{-3}} = 9000 \qquad (53)$$

ie RATIO 9000:1 !

or normalised to seconds, 1 second of VOSTTAC1 time is equal to $2^{1}/_{2}$ HOURS OF BBC TIME!

It is also well worth noting that the slave TMS32010S can be performing some other process during the 40mS. Forgetting for the moment numerical ratios and rates, and resorting to layman terms, edge enhancement at different orientations can be done sufficiently quickly to enable through line displays if required.

Whilst on the subject of the DTP, it should be noted that the template prom is not necessarily restricted to edge detection and averaging. It could, if suitably programmed, be used as a simple local thresholder, palette, region grower/shrinker and hole filler. One further usage perhaps worthy of consideration is that of the separable median filter. This is a non-linear RANK filter which is simple to implement and approximates to the 3x3 MEDIAN mentioned earlier. The first pass would replace a pixel by the median value compared to two neighbours, and the second pass would be a repetition of the first but scanning in an orthogonal manner.

6.9. COLOUR EDGES

The use of colour has made very little impact on edge detection despite its success in the dual or region based techniques. The main reason for this is that luminance and chromatic edges tend to co-occur. This was investigated in some depth by (ref 77). The reasons for this are not hard to find. Firstly, regions of different colour are generally unlikely to have the same luminance value. Secondly, from a practical point of view the luminance information in the PAL TV system has up to 5 times the bandwidth of the colour information, ie 5MHz:1MHz ratio for a good quality system. .As a realistic example consider a low quality source with

a 2.5MHz:1MHz ratio applied to VOSTTAC1. A 2.5MHz signal would have a period of 0.4US and with 384 samples per line (sample period of 0.135US), approximately 3 samples would span the fastest change possible. For a colour change only, the minimum period is 1US for which 7 samples are required. Now because only small templates are used (3x3) the maximum output occurs when a very steep gradient is under the mask. Therefore luminance changes will dominate the output as colour is a much slower ramp only really noticeable over considerably larger templates. This type of performance is analogous to the eye for which it is suggested that colour is used for identifying areas rather than discontinutities.

Colour edges can be used to improve the confidence of an edge detection, i.e. by observing how many colour edge spaces agree which can assist the segmentation of low contrast images. Because of this VOSTTAC1 provides the facility to edge enhance all 3 colour spaces so that subsequent processing has every processing option available. It is worthy of comment that if a B/W source is used within the colour system or if the colour is "killed" within the decoder each colour space represents a copy of the luminance signal.

6.10. ENCODING EDGES

Having introduced various methods for edge enhancement and discussed practicalities and speed constraints edges would seem to be fairly well covered. Certainly from the point of view of many

applications this would seem acceptable, but for others it is not. What exactly does an edge transform achieve? It is merely a means of highlighting pixel level discontinuities. Whilst this is often all that is required for further human interpretation it contains only local information about any edges, ie no knowledge of estimated average loci. So for processes which are rather more sophisticated/intelligent than a simple enhance, a strategy is required for coding the available information. For example it is almost impossible for a computer to make some decision about many local edge results before combining them into some mathematical form such as a line or curves (CONICS). To the uninitiated the task of producing a contour description of say a straight line, might seem trivial, but a glance at the outputs of PIC (42) should dispel this idea. Even apparantly regular lines or in this case roads, once edge operated, are likely to have gaps, noise and deviations from the norm. A contour following technique is not generally applicable, true small gaps can be traversed and small deviations allowed for but more dramatic "noise" effects are likely to foil the efforts of moderately "dogged" followers, and of course the more sophistication used, the slower the technique. If the problem is considered as a graph fitting exercise, the least mean square fitting criteria would spring to mind. This is a perfectly valid technique in theory, however its high speed practical application is lacking. In its usual form a key point of the analysis is to invert a matrix. Applied generally the rank

of this matrix can be quite enormous depending on the number of pixels set by the edge enhancement and therefore the minimisation proces is slow.

A technique becoming increasingly popular within signal processing circles, although perhaps less well known amongst mathematicians is the Hough Transform. Originally applied to examining paths in bubble chambers (ref. 78) the scheme has been popularised by the work of Duda & Hart (ref. 79). Basically the algorithm systematically considers all lines within a specific range passing through all points and assesses the popularity of each. The general method is as shown in FIG (36).

A problem with this method is that if all M (slope) in coordinate space is considered, C (intercept) may became infinite. A modification is to enhance along several gradients in the region under consideration and perform the Hough transform based about the most popular scan angle. In this way the sought after best fit line will only deviate over a small sector, ie +/- 30^{o} say for horizontal/vertical and diagonal scans. Note applying the Hough transform along a 45^{o} orientated box is only possible using vector scanning edge detection and transfers, facilities peculiar to VOSTTAC1.

Experimental tests were conducted using a terrain model to represent an aerial view of a scenario. For the picture

considered the 45° version was chosen as the dominant orientation, see PIC (42). The test area was constructed from two windows, for which accumulator arrays were constructed and overlaid on the original image, see PIC (43). Red "spots" represent local maxima and hence line equations from which the blue lines were drawn. The Hough technique is applicable to curves as well, the differences being the equations for computing the array contents and the order of that array. Although the Hough method is a fairly resilient line finder it is not ultra quick, depending on the number of active detected pixels, and the definition accuracy of the accumulator array. Despite this, it is an engineering solution to a tricky problem, by that it is meant that it can be applied by a regular process suitable for hardware implementation. Therefore for applications which would use this extensively the firming up of the algorithm may well be justified.

Edge enhancement is a discipline in itself but performance comparisons with region techniques must be made after the coding process is complete eg Hough transform or some other boundary finder (refs 79,80,81). The main conclusions may be tabled by the following qualitiative observations.

6.11 EDGE V REGION

1) Region segmentation guarantees closed boundaries, edge based techniques may only yield sections of boundary, and unless a very good (and usually slow) line follower is used, the supposed boundary may diverge uncontrollably.

2) Edge detection is usually a local operation and hence local or short span failures may prevent adequate segmentation, even when a boundary is apparently obvious to the human visual system. Region segmenters are more global, so less sensitive to and more likely to miss low contrast small objects. This may be improved by spatially variant techniques, but there are occasions when region techniques are inapplicable. For example the snow tracks of PIC (44) would have very little influence on global histograms, but show up sufficiently well when edge enhanced to allow Hough coding (note tracks veer off to right, as shown by mean line equation).

3) The use of colour information can greatly increase the performance of digital region segmentation techniques, as colour tends to be an area attribute. This is analogous with the performance of the eye which also deals with colour as a low frequency or region based property. Conversely the inclusion of colour has relatively minor effect on edge detection which deals with high frequency discontinuities which are normally present in the luminance signal.

4) The position of detected edges is relatively insensitive to global parameters such as threshold. However the position of region boundaries can be quite sensitive to the choice of range for the segmentation attribute.

As far as real time operation on VOSTTAC1 was concerned, the conclusions were as follows:

a) High rate tracking should initiate via a mainly region segmentation technique

b) Multispectral thresholding is a powerful and fast region segmenter

c) Edge enhancement is a fast technique using the DTP

d) Edge enhancement may assist or limit a fast region segmentation process

e) General edge detection can be used for less rigorous time constraints, ie towards off line analysis unless special hardware added

Whatever methods are used for isolating sub areas of interest, further attributes are required to distinguish between valid and invalid regions. Area is the simplest to calculate and often a byproduct of the segmentation process. For single dominant regions in the presence of noisey interference this may well be adequate. However, there is the likely situation when objects of similar size present themselves as target candidates. Perhaps the most telling of static comparisons is that of shape and a means of producing suitable measures is discussed in the following chapters.

CHAPTER 7

SHAPE DETECTION AND CODING

7.1. SHAPE

Shape is a very well established concept to the human observer, being used for a multitude of real world descriptions. It is not however an attribute which is inherently obvious to digital image analysis and can only be found by a series of tests and encodings (ref. 82). In a local sense the perimeter of an object may be described as a set of contiguous but variably orientated line segments which as a group represent the entire boundary. The processing problem is then how to encode this information into a convenient and computationally economic descriptor, suitable for comparison/decision making. One approach is to produce a chain code (ref. 83). Essential to this process is that the boundary be noise free, which then allows a small directional mask or "bug" (refs. 84, 85) to be passed around it as shown in FIG 37. At each boundary point the mask is used to test the angle between the current and next position, and this is coded as the relevant chain code number.

7.2. THE CHAIN CODE

By starting from an arbitrary boundary position the perimeter is followed at each stage identifying the relevant code and adding it to the list or "chain" until the initial point is reencountered. Once collected the chain code is a complete representation of the shape.

Consider for example the closed curve of FIG (37a). This is specified by the chain code 3221066664 from which any geometric property should be calculable. One desirable process might be to find the enclosed area (actually = 7 by simple observation).

Freeman has shown that various codes affect area in following way:

CHAIN CODE RULES

Scope	Change in Area	Change in Modifier
0	+B	+0
1	+B+1/2	+1
2	+0	+1
3	-B-1/2	+1
4	-B	+0
5	-B+1/2	-1
6	+0	-1
7	+B-1/2	-1

AREA CALCULATION SEQUENCE FOR THE EXAMPLE OF FIG 37

N	CODE	A	B
0	3	$-1\frac{1}{2}$	2
1	2	$-1\frac{1}{2}$	3
2	2	$-1\frac{1}{2}$	4
3	1	3	5

4	0	+8	5
5	6	8	4
6	6	8	3
7	6	8	2
8	6	8	1
9	4	7	0

The modifier is an offset which in this case would affect the integration along the X axis. For the example considered the modifier (B) is = 1, as the start point is one Y unit from the X axis.

Using the chain code rules it is possible to calculate the sequence shown above by a methodical process and hence also the area.

A much more detailed treatment is given in ref. 86, dealing with manipulation codes for object, expansion, contraction, rotation, curve length, inverse digits, path reversal, path reduction, closure, multiple tap curves, intersection and the area determination previously mentioned. Generally the chain code is a useful representation and a good stepping stone to the high level analysis, i.e. complex pattern recognition. However the method is not ideal as a high speed matching criteria.

As already hinted the problem with this technique and boundary followers in general is their sensitivity to noise, bearing in mind that such algorithm noise immunity can be increased only at the expense of speed. In addition without further processing the code representation itself is not always a very compact description of the object.

7.3. TEMPLATE MATCHING

Overcoming these obstacles makes the technique slower than it at first appears. As far as identifying targets in the presence of noise in real time is concerned, some quicker and more robust strategy is required. From the field of real time target tracking, the most well known and extensively used shape comparator is the template correlator. Although this is a very simple method its performance has been hardly rivaled for high rate processing and can be found in ultramodern equipment such as the British Aerospace visual multitarget tracking system (ref. 65).

Basically the technique proceeds as follows:

a) Capture and learn a window centred on a known target to generate a model

b) Correlate a new target candidate on a pixel by pixel basis, for various offsets between actual and model grids

c) If match greater than fixed level then adequate correlation

Because the correlation is a regular addressing method it is directly applicable to sequential data processors such as the TMS32010 or more importantly to dedicated hardware. Consequently for modestly sized windows the comparison can be extremely fast and hence its popularity for real time tracking systems. However this method is not without drawbacks, which enables its position as the quickest, most elementary shape comparator to be challenged. The disadvantages may be listed thus:

1) Models are not compact. Stored as individual pixel values these can require a singificant amount of memory especially if several models need to be held.
2) Computed models are of fixed size. If the object is noticeably expanded or contracted perhaps due to a change in normal distance from the sensor axis, then the template match is unlikely to "trigger".
3) Models are of fixed orientation. Even a modest change in orientation can ruin the matching process.

The importance of Point (1) will largely depend on the practical implementation, but even large address spaces can soon be used up storing sub-pictures (recall that a typical digital display requires more than half a megabyte of memory!). To be fair, templates do contain much more information than just shape so the

storage of recent models may well be justifiable if practical. Points 2 and 3 are more serious and various attempts have been made to overcome these shortfalls using different sized and pseudo orientated templates, which gives an unwieldy and inelegant solution. All this is very conflicting with the human idea of shape. For example if a person was asked to describe a beer mat lying randomly on a table, the consistent description would be that of a square, say and not further qualified by size or angular measures (provided of course that the person was sober!). A conceptual approach which fits the accepted notion of shape makes use of moments. This area has attracted much attention pioneered by Hu (ref. 87). The technique involves taking radial measurements from boundary positions and using them summed to various powers to compute a set of invariant moments.

7.4. INVARIANT MOMENT

The general discrete equation for moments is:

$$u_{pq} = \frac{1}{N} \sum_{i=1}^{N} (U_i - \bar{U})^p (V_i - \bar{V})^q \tag{54}$$

From which a set of moments invariants can be defined:

$$M_1 = (u_{20} + u_{02}) \tag{55}$$

$$M_2 = (u_{20} - u_{02})^2 + 4\, u^2_{11} \tag{56}$$

$$M_3 = (u_{30} - 3u_{12})^2 + (3\, u_{21} - u_{03})^2 \tag{57}$$

$$M_4 = (u_{30} + u_{12})^2 + (u_{21} + u_{03})^2 \tag{58}$$

$$M_5 = (u_{30} - 3u_{12})\,(u_{30} + u_{12})\,[(u_{30} + u_{12})^2 - 3(u_{21}+u_{03})^2] \tag{59}$$

$$+(3u_{21} - u_{03})(u_{21}+u_{03}) \quad [3(u_{30}+u_{12})^2 - (u_{21}+u_{03})^2]$$

$$M_6 = (u_{20} - u_{02})) \; [(u_{30}+u_{12})^2 - (u_{21}+u_{03})^2] + 4u_{11} \; [(u_{30}+ u_{12}) \; (u_{21} + u_{03})] \qquad (60)$$

$$M^7 = (3 \; u_{21} - u_{03}) \; (u_{30}+u_{12}) \; [(u_{30} + u_{12})^2 - 3 \; (u_{21} + u_{03})^2] \qquad (61)$$
$$-(u_{30} - 3u_{12}) \; (u_{21} + u_{03}) \; [3(u_{30} + u_{12})^2 - (u_{21} + u_{03})^2]$$

Dudani et al (ref. 88) used these proporties to classify aircraft under varying conditions of view aspect i.e. azimuth and roll angle. In that work Bayes, or distance weighted nearest neighbour classifications were used. An important result was that although the matching process was not 100% accurate, it performed much better at classifying particular binary profiles than a human observer. The investigation was offline and a particular matching process took 30seconds. Problems associated with the algorithm were the sluggishness of the statistical classifier especially for large model sets. Although the paper suggested that special hardware could be produced to perform at frame rate it would be unlikely that this could be achieved without significant changes to the algorithms. For example it is easier to measure boundary distances of a blob, as these have closed boundaries whereas edge techniques struggle to guarantee this at frame speeds.

Another more important drawback as far as high speed operation is concerned is that searching for the radius measurements is not something naturally achieved using a cartesian screen map, i.e. the technique is exclusively in polar form and therefore converting back and forth from X&Y co-ordinates is unwieldy. What is then required is some rapid means of sampling length along a number of orientations. Furthermore those measurements should then be normalised to a signature, from which model comparisons may be made irrespective of size and orientation, yet yielding both these parameters.

7.5. A HIGH PERFORMANCE SHAPE DESCRIPTOR FOR THE VECTOR SCANNING MACHINE

Before investigating the detailed production of the measurement scans or "spokes" it was necessary to ask a question. Assuming an ideal polar co-ordinate system, could a technique be developed, suited to the TMS32010s which would allow differentiation between standard shapes. To investigate this a suite of off-line shape descriptor simulation programs was produced. A number of common shapes was used as a test set (FIG (38)) including a square, parallelogram, bar, triangle, hexagon and trapezoid.

After radial measurement of a shape the first necessary process is to normalise these values, to divorce the Euclidean distance measures from their actual scale, e.g. pixels, centimetres, inches etc, and yet present the same result set irrespective of object size. The method of scaling and output range produced will depend a great deal on the subsequent matching criteria, and therefore

this must be decided. Considering that the measurements represent a periodic 1-D waveform, the TMS32010 could quite easily perform a correlating technique for model matching at high speed. Therefore a suitable scale might ensure that the peak of the autocorrelation function be unity. Before this can be considered the validity of the correlation process must be questioned. If the technique is applied to a shape by using the radii measures directly, many shapes would produce a triangular cross correlation function. This is because the measurements have a d.c. offset which for blob like objects makes the signature appear as that characteristic of a square wave, due to constant radii over limited range. It is therefore essential that only the contour fluctuations or a.c. signal component is analysed.

Therefore the measures should be defined more precisely, as the difference between the actual boundary and that of a circle of equivalent area, normalised with respect to the mean radius.

e.g. $$R_{AC} = \frac{R_{MEAS} - R_{MEAN}}{R_{MEAN}} \qquad (62)$$

R_{MEAS} = Measured Value

R_{MEAN} = Mean Radius

R_{AC} = A.C. Components

It is still not possible for these components to used for successful cross correlations, as no consideration has been given to the varying power contents of measurements. If this is ignored crosscorrelations of differing objects may well give greater output response than the autocorrelation function. For example the autocorrelation of a low power boundary such as a hexagon would be less prominant than a cross correlation with a powerful function such as a bar say. So what can be done to ensure adequate and consistent scaling?

Consider then the general expression for the cross correlation function:

$$R_{fg}(\tau) = f(t) * g(-t) = \int_{-\infty}^{\infty} f(t)\, g\,(t+\tau)\ d\tau \qquad (63)$$

The autocorrelation is then simply,

$$R_f(\tau) = f(t) * f(-t) = \int_{-\infty}^{\infty} f(t) f(t+\tau) d\tau \qquad (64)$$

Properties of this function are:

a) Always even

b) Is a maximum at $\tau = 0$

c) $$\int_{-\infty}^{\infty} Rf(\tau) d\tau = \left[\int_{-\infty}^{\infty} f(t) dt\right]^2 \qquad (65)$$

Using property b

$$\text{MAX } R_f(\tau) = R_f(0) = \int_{-\infty}^{\infty}(f(t))^2 dt \qquad (66)$$

Normalise such that $R_f(0) = 1$

Then

$$R^o f(\tau) = \frac{f(t)*f(-t)}{\left(\int_{-\infty}^{\infty} f(t)^2 dt\right)} = \frac{1}{\left(\int_{-\infty}^{\infty} f(t)^2 dt\right)} \cdot \int_{-\infty}^{\infty} f(t).f(t+\tau)d\tau \qquad (67)$$

As scalling of the input waveforms is of interest (f(t)) then scan scale thus

$$R^o f(\tau) = \frac{f(t)}{\left(\sqrt{\int_{-\infty}^{\infty} f(t)^2 dt}\right)} * \frac{f(-t)}{\left(\sqrt{\int_{-\infty}^{\infty} f(t)^2 dt}\right)} \qquad (68)$$

(where $\int_{-\infty}^{\infty} f(t)^2 \; dt$, is readily calculated).

Therefore measurements should be normalised to one over the root of their power content.

This secondary normalisation ensures that cross correlation of dissimilar targets are less dominant than their respective autocorrelation functions which may peak at unity.

Preliminary simulated tests yielded the waveforms of FIGS (39a-41d). Note that the position of the dominant peaks actually occur at the X origin but are shifted for display purposes.

Scaling in the way mentioned produces shape signatures that are size independant and yet produce the area as a byproduct. Should the target become skewed, a sliding to, over and past, type of correlation is no longer valid and so a cyclic variant is used, i.e. one waveform is "rolled" around the other so that at any shift value every pixel in one waveform is covered by one from the other. This is possible as the signature is periodic, and ensures that the model and signature have every opportunity to be correctly aligned.

Referring to FIGS (42a-43b) the cyclic autocorrelation of a number of shapes can be seen, i.e. hexagon trapezoid triangle parallelogram, square and bar. Note that due to multiple axes of symmetry, some shapes produce multiple dominant peaks, which is an important descriptor in itself. Various cross correlations are shown in FIGS (43c-47a). Note the generally diminished peaks relative to the autocorrelation functions. However one comparison is not particularly rejective, i.e. the bar against parallelogram has high peaks. This is not totally unexpected as in the limiting case as the parallelogram is "squashed", It becomes very bar like, i.e. two axis of symmetry, and similar aspect ratio due to one

dominant direction. However the simulated comparison was limited by too low a sample rate of 32 measurements, which meant that the peaks of the parallelogram were missed. To a lesser extent this is true for the case of a triangle and trapezoid, which if undersampled appear very similar. Again this is not surprising as the trapezoid is identical to the triangle but with one of its peaks knocked off. As an enhancement it was decided that a practical system should have at least double the angular sample rate i.e. 64 as opposed to 32 measures.

Having proposed a means of comparing objects of dissimilar size and orientation, which also generates area, what then of the actual target skew? This value is in fact inherent within the correlation function. When no change in orientation is relevant then the auto correlation waveform will have a dominant peak about the origin, regardless of the type of shape. When the target becomes skewed relative to the known model, this peak begins to shift from its initial position. Therefore by measuring how far the nearest peak is from the origin (bearing in mind that the waveform is periodic) the number of angular increments or decrements can be found and hence the approximate skew can be estimated. The measurement can only be defined to the nearest sample scan, which for a 64 point function gives the angular increment shown

$$\theta INC = \frac{\text{SECTOR SIZE IN DEGREES}}{\text{NUMBER OF SAMPLES}} = \frac{360^{o}}{64} = 5.625^{o} \quad (69)$$

Therefore a plausible technique was proposed for shape coding and target identifying irrespective of size and orientation yet yielding these parameters.

7.6. PRACTICAL POLAR COORDINATE MEASUREMENT

At the beginning of this discussion a big assumption was made, that an ideal polar co-ordinate scanning system existed, which for conventional hardware is certainly not the case. Infact usually only the X and perhaps Y scans are easily available which would give a θINC of 90^{o} and so only a crude measure of aspect ratio (ref.89) could be gained. To measure radii along a properly orientated polar vector is very easy but having to produce angular calculations and searches using an XY grid addressing scheme would be a great overhead to the process. What is then required is a method of presenting the vector information as a sequential data stream. It would then be a simple matter for the TMS320S (or hardware) to search for the valid boundaries radii perhaps with the option of a simple 1-D noise filter. If this cannot be achieved then the technique has little merit. VOSTTAC1 has been heralded as a high speed vector scanning image analysis machine, but its ability to produce fairly precise line scan orientations within a finite medium resolution bit mapped display was not initially certain.

In Chapter 3 the major modes of the VOS machine were discussed i.e. LINE & BLOCK modes. Block transfer is clearly inapplicable to the shape measured as precise angular definition is usually associated with increased interpixel distances and therefore radii estimates could be inacurate. Furthermore transfering blocks or windows involves moving a significant amount of data, the majority of which is redundant. Line mode is the logical alternative, which uses convenient x,y ratios to build orientated vectors from smaller line segments.Of course there are associated errors with such an approximation but providing the ratios (and offsets) are carefully chosen, the maximum deviation from the mean vector can be kept below 2 pixels (on average coincident with vector) and the interpixel distance weighting relative to the horizontal sample rate less than 2. On this basis it was possible to construct a driver table to produce the 32 different scans (2 measurements per scan about the origin). The calculation of the tabular coefficients was no trivial affair. Converting scans to ratios and then to increments/decrements, offsets and then finally to 1's complement driver values was sufficiently involved to merit the creation of off-line programs to perform the tasks automatically. This strategem also took precautions against violating the frame stores Pseudo-Random access. Once computed the driver table allowed the required vectors or "spokes" FIG 48, to be produced and generated the required vector line sequential data streams. The radii search is then trivial and so to is the primary

normalisation (i.e. calculation and subtraction of mean radiis), however the secondary scaling presents a problem. Recall that the scale factor requires the square root of a power measurement and due to the nature of the boundary waveforms combined with the sum of squares operation, this factor may vary over an enormous range. It is therefore very difficult to maintain high accuracy using fixed point techniques, and the problem is beyond the scope of a small function look up table. A variable with a large dynamic range is a classic candidate for floating point representation for which square rooting is relatively simple. Using such a representation in a high speed system would seem to be going against the "grain" or philosophy of approach especially with no floating point co-processor available for the TMS32010. But infact it is not such a reckless sellout to convenience, because it would only be used sparingly, i.e. one number normalized, rooted and then renormalized. Providing the final result is then converted back to fixed point prior to scaling the technique is not too slow. It is therefore worth remembering that even the most obviously sluggish techniques should not be dismissed out of hand, if only used sparingly. Once finally normalised the actual correlations are quick, easy and efficient. If a new object is compared with a set of several models, the best fit can be found by the largest integral of the cross correlation functions. Once matched the dominant peaks are identified and recorded and the peak shift measured to estimate the skew.

Experiments on VOSTTAC1, proceeded by first learning a knowledge set of 4 targets physically situated at the bottom of the screen. The next stage was to nominate a target (central screen) to be correlated with the model set. The best match model would then be indicated, the peaks counted, percentage magnitude calculated, and skew estimated. Although VOSTTAC1 is not designed to produce computer generated graphics its primary display was used to output the results simply for convenience. Practically the graphical and textual output was achieved by modification of the contents of a small window and then zoomed to the required size and position on screen. PIC (45) shows a square target with its model set. PIC.(46) gives the cross correlation waveforms with models 0 to 3, and text indicates the decisions made, i.e. shape 2 = square, 4 peaks, 0^{o} skew (PK.1) = 65%, (PK.2) = 77%, PK.3 = 61%, PK.4 = 93% (where PK implies percentage magnitude of peak). PICS (47) and (48) are similar but for a diagonally positioned square, yielding similar matching but identifying the skew as 45^{o}. Similar results are shown in PICS.(49-56) for a bar and triangle. The circle PICS (57-58) are a special case.

Recall that the crux of test is to measure the deviation of a boundary from that of a circle of equivalent area. Therefore discs can be identified at an early stage by a noticeable lack of perimeter power, subject to some limit. In fact the choice of this threshold affects the overall sensitivity of the test and is a compromise between detecting low power polygons, (i.e. octagons

and above begin to resemble circles) and yet preventing noise on circular boundaries from being blown up out of proportion. For example, without a threshold and applied to near circular shapes, the process becomes a noise correlator.

7.7. SPEED OF SHAPE COMPARISON

Having postulated a technique, implemented it on a machine, the next question is how fast is it?

The only way to answer this is to give a representative example. Consider then the main sections of the procedure (ref. 22) with relevant routines shown in brackets.

A SIMPLE TARGET CENTRE FIND

(TGTID)

B VECTOR SCAN LINE DUMPING

(VCINIT, GETVSC, DUMP-TIME)

C RADII MEASUREMENT AND PRIMARY SCALING

(NRMEAS, RSTRE, SMEMRW, RDNRM, RAVGE, DCOFFR)

D SECONDARY SCALING

(SCLCOR SCALEB SIGSTR)

E CORRELATIONS WITH STORED MODELS

(CYCCOR)

F BEST MATCH POWER INTEGRATOR WITH STORED MODELS

(CORPOW)

G BEST MATCH WAVEFORM ANALYSIS

(BGSRCH, PEKABO, CORINF)

Sections A, B, C, D, and G execute once per test but E&F are repeated depending on the number of models in the knowledge set, i.e.

```
DO        A
DO        B
DO        C
DO        D
FOR I = 1 TO N (Where N = Model Count)
DO        G
DO        F
NEXT I
DO        G
end
```

Relevant approximate timings for individual routines based on the square example are now given:

ROUTINE		TIME	CYCLES
A	TGTID1	20 mS	
B	VCINIT		47
B	GETVSC		341
B	DUMP-TIME	1 mS	
C	NR MEAS		728
C	RSTRE		15
C	SMEMRW		403
C	RDNRM		1063
C	RAVGE		325
C	DCOFFER		451
D	SCLCOR		1207
D	SCALEB		583
D	SIGSTR		787
E	CYCCOR		31,171 PER MODEL
F	CORPOW		349 PER MODEL
G	BGSRCH		45
G	PERABO		1325
G	CORINF		572

Assuming a 200nS cycle time T

A = 20ms

B = 1mS + 354 (T) = 1.1mS

C = 2985 (T)

D = 1497 (T)

E = 31171 (T) PER MODEL

F = 349 (T) PER MODEL

G = 1942 (T)

The total time may be considered as the sum of a constant and a variable weight constant, i.e.

$$TOTAL = K_1 + NK_2 \quad (70)$$

Where N = number of models to be correlated with and

$$K_1 = (A+B+C+D+G), \quad K_2 = E + F \quad (71)$$

K_1 = 22.5mS K_2 = 6 mS

Therefore assuming Total represents a frame time of 40mS, then the target could be found using the simple identifier and then matched against up to 3 models in a knowledge base. To say the technique is real time (image rate), fast is then well justified and is certainly more eligant than the conventional template matcher. It also scores by requiring considerably less storage for its models. Even though to speed the correlation process the models are stored twice in memory.

Therefore for a 64 point function 128 words are required so 16 models could be stored in 2K. A picture template by comparison based on a 90x90 grid (similar range to VSCAN method) would

require 8K x 16 i.e. 131072 words to achieve this. Therefore not only is the VSCAN technique phenominally quick but it requires less storage than direct template matching by a factor of 64.

It should also be appreciated that although the TGTID1 was a double slave sequence, the remaining timings were computed for a single slave sequence so if required even further speed ups should be possible by better use of slave parallelism especially for the correlation processes. Ideally four slaves could match up to 12 models in a frame time.

Furthermore if the centre was assumed, perhaps in some inspection system the theoretical maximum number of model correlations would be:

$$\text{INT}\ (40/6)\times 4 = \underline{24} \quad \text{(4 slave)} \qquad (72)$$

or 6 for single slave machines.

From the foregoing considerations VOSTTAC1 is extremely adept at this kind of shape comparison, having reduced a complicated spatial random access problem to a number of operations on sequential data streams. For blob like or polygonal shapes the technique works well, which means it is applicable to a large number of real world targets, i.e. houses, tanks, ships etc. But no technique is infallable and simple application of the shape coder to some partially ocluded or complex objects can sometimes result in ambiguity.

For the case shown in FIG 49 the two shapes will have similar boundary signatures although they are unlikely to be exactly the same as the centre of area is likely to shift slightly.

7.8. HIGHER DIMENSIONAL SHAPE TESTS

Various strategies could be attempted to encode more complicated or less blob like targets using slightly different strategies. If more precision was required near the boundaries an array of spokes could be used as in FIG. 50.

If the primary scan (0) seems acceptable then secondary scan/tests 1-8 can be attempted for more definition.

A similar variant is possible for holes within a major region (target). Having encoded the target, a further stage might code the boundary of the secondary region and store it as a secondary attribute.

The technique is also extensible to 3-D. A problem with 3-D (ref. 90) is that a comprehensive set of view models require a large amount of storage and hence the matching processes are very slow. To compensate for this usually only a few snap shot templates are used, but these suffer from the usual problems of fixed size and orientation, and still require appreciable memory usage.

Therefore the VSCAN code is ideal, i.e. very fast at matching and stored models are compact.

SNAPSHOTS	1 SLAVE	4 SLAVES
6	80mS (within 2 frames)	40 mS (within 1 frame)
$8^3 = 512$	3.07S	0.77S
$64^3 = 262144$	26 Minutes	6.5 Minutes

(Six is probably practical number the last entry is included as it would give the same angular accuracy as the 2-D test in all 3 dimensions).

Storing large model sets is not necessarily essential if say a tracking system is initiated by a user nomination function. In this case the first shape model is learnt immediately on-line and so the initial best fit comparisons are avoided. Subsequent operations then reduce to a means of comparing the current shape to the previous one and if acceptable matching subject to a threshold then tracking is maintained and a relearning model adaption would be performed to ensure that the most up to date representation was available. Refinements might be to keep a small number of models on a small FILO stack to increase confidence when matching is questionable and perhaps to keep a fixed best approximation model to use for re-identification should tracking fail.

To summarise then, a very high speed algorithm has been implemented on VOSTTAC1 which fits the human idea of shape. The use of vector scanning algorithms on VOSTTAC1 has been mentioned in two publications, ref. 21 and ref. 22, and roughly related work can be found in refs. 91 and 92, and program sources in ref. 45. The test essentially measures variation of the boundary from a circle of similar area, by measuring radial distance from the blob, or target centre. These measurements are reduced to tests on 1-D sequential waveforms by extensive use of vector line scanning. By scaling these values relative to the peaks of their autocorrelation functions, waveforms signatures become irrespective of object size and can be matched against models using cyclic or periodic correlations so that the comparison is also angularly independant. Furthermore the area and skew parameters are produced as a byproduct of the correlation process. The strategem is extremely fast (typical example for matching 6mS) and yet stores the boundary information in a compact form. The technique would seem expandable to higher dimensions.

In this chapter and in fact throughout this thesis references have been made to target tracking applications but as yet no real discussion had been given to VOSTTAC1's ability to tackle such problems. For this reason the following chapter reviews some tracking activities investigated using the rig.

CHAPTER 8

TARGET TRACKING

8.1. TRACKING BY COMPUTER

Target tracking in various guises has been of great interest for several decades, especially when applied to radar and infra red sensing. The principal motivation being a desire to produce military equipment such as early warning and terminal guidance (ref. 65,66) systems. As a direct consequence, much of the related research work is subject to publication restrictions. Coupled with the fact that very high speed tracking (video frame rate say) by digital computer is still uncommon, there is conspicuous sparsity of information in this area of image analysis.

8.2. SCOPE OF INVESTIGATION

The majority of universities are limited to evaluating captured images off-line, primarily due to a lack of specialised frame rate processing hardware. VOSTTAC1 was motivated by a desire to remove this restriction in order to gauge the effects of the algorithmic and practical problems associated with high rate analysis including target tracking. However it was not intended that the MKI prototype should function as an expert system encompassing strategic tracking decision making, but rather as a low knowledge object tracker. This is the most critical performance level which will dictate the minimum cycle times for any VOSTTAC TMS32010 based system. Satisfaction of time constraints would not only consolidate the design of VOSTTAC1 but also justify its evolution

to later versions. What is then of interest is not initially the evaluation of complex target loci histories but the dynamic performance of the fundamental tracking hardware and algorithms, discussed below.

Consider an elementary tracker in order to make some predicitions and measurements of its dynamic performance. The most favourable conditions would prevail, realised as a binary image, with a single active target region.

The associated loci could then be monitored as a succession of blob central co-ordinates relative to a 90 x 90 grid (coarse resolution). The actual midpoints can be yielded by a two slave, two pass concurrent process. The first pass of which proceeds by scanning window lines and computing an area sum for each, so that on completion of this phase a 1-D array of partial results is produced, along with a total area count. The next step is to search the array until the sub areas either side of the array pointer are approximately equivalent and valued at half the total area. In this fashion a central co-ordinate is generated relative to the window dump scan angle. Therefore by downloading horizontal and vertical scans, to two separate slaves and performing the same test on each, both X & Y co-ordinates are produced simultaneously. The procedure has the following properties:-

1) Noise averaging - due to area being used as a centring criteria, co-ordinates are resilient to low area random noise.

2) Works for any scan angle. - note that the programs for calculating X, Y or any other angle measure (say $\pm$ 45°) are exactly the same, it is only the physical dump angle of the data that differs.

3) Maintains sequential data streams - VOS maintains this format and therefore minimises addressing overheads which is essential for fastest possible operation.

4) Both window scans performed in the 1 dump time - this makes use of the Orthogonal scan facility of VOSTTAC1 generating horizontal and vertical (or any suitable orthogonal pair) scans simultaneously.

From the foregoing considerations it should by now be clear that the approach is tailor made for VOSTTAC1 and should therefore be efficient and hopefully quick. To define the performance less vaguely requires first that the critical signal processing loop of the program be subject to scrutiny, to yield speed limitations.

8.3. TRACKING LIMITS

An adequately precise model of the overall tracker processing time may be written as follows:-

$$T = D + S + R \tag{73}$$

where

T = TOTAL PROCESSING TIME

D = DUMP TIME _ 1ms

S = MEMORY SCAN TIME = NxNxMxC (74)

R = RESULT SCAN TIME = NxPxC (75)

M = AVERAGE INSTRUCTIONS PER PIXEL = 10

P = AVERAGE INSTRUCTIONS PER RESULT = 10

C = INSTRUCTION CYCLE TIME = $200x10^{-9}$S (5MIP)

N = RANK OF WINDOW

N.B. (Although M & P seem convenient they are measured and not estimated values!)

For a rank 64 x 64 window the processing time is:

$$T = 1ms + (64)^2x10x200x10^{-9}+64x10x200x10^{-9}$$

$$T = 9ms \tag{76}$$

For a rank 90 window:

$$T = 1ms + (90)^2x10x200x10^{-9}+90x10x200x10^{-9}$$

$$T = 17ms \tag{77}$$

Therefore tracking using a 90x90 co-ordinate grid within a frame period (40ms) is in theory easily attained and even field rate operation (20ms) is conceivable. The co-ordinate grid is obviously much reduced relative to full screen resolution, i.e. YSIZE/8 BY XSIZE/4 which is equivalent to having an image similar to PIC (3). For tracking purposes this is sufficiently fine, the trend being to use the higher resolutions at local level when required in an area specified by some coarse blob co-ordinate. In this way speed is not wasted accessing redundant data. By analogy one might regard the screen map as a filing system. Examining the picture at full resolution directly is equivalent to searching a large sequential access file, which is renowned as a time consuming process. (Should not be confused with the sequential data streams produced by VOS which give high speed signal processing.) By comparison the provision of a coarse blob co-ordinate is much the same as an index to a part of a sequential file which characteristically speeds the search dramatically (eg ISAM files). What should be appreciated then is that initial tracking at the early coarse resolution chosen is not only acceptable but in fact provident.

Pen and paper calculations are all very well, but they are usually only indicators to what might be achieved in practise. Real world systems involve a great many variables and external conditions, including I/O constraints, event handling bottlenecks, data transference, etc. of which only the major effects are modeled. As a consequence, simple predictions cannot be a 100% accurate.

Where practicable it often requires less effort to substantiate performance claims by measurements on a running system, than deriving a comprehensive model for it.

8.4. EXPERIMENTAL CONDITIONS

Defining experimental conditions for the dynamic tracking measurements, extended the binary image tracker mentioned previously to an intensity follower, i.e. looks for lightish objects on a dark background. This allowed the test image generator of PIC (59) to be used, which is a thinly disguised and upended chart recorder. Motional control is then simply accomplished by exciting the X&Y drive inputs with low frequency oscillations. The input to VOSTTAC1 being only visual and via a low cost video camera. The results produced by acting on this data could be monitored by one or both of the two output mechanisms provided, either visual monitoring of the auxiliary display tracking cross or by recording of positional and velocity signals from the four O/P DAC channels.

The visual indicator is not easy to convey with still photographs, but careful examination of the time lapsed loci of PIC (60) should reveal a number of tracking crosses superimposed on the target path. The analog outputs are a more measurable output and during the experiment were fed to a multichannel U-V trace recorder after appropriate buffering (using an analog computer). In this way output traces could be created from a target moving at specific controlled frequencies.

With the intention of phase measurement it was invalid to compare the input to the chart recorder with VOSTTAC1 output signals due to the likelihood of delay within the recorder itself. To surmount this obstacle, the reference input measurement was taken from the positional feedback potentiometers of the target controller.

A copy of a typical trace is exhibited in FIG (51) from which it can be seen that there are 25 outputs per one second marker, i.e. frame rate tracking (40ms). Moreover, examination of slave BIO lines which represent their relative activity revealed that the process was indeed complete within 20ms, the remainder of the cycle being a wait for new frame. This bears out emphatically the speed of the system and also the relevance of the earlier performance estimation. This would seem to suggest that tracking could be achieved at field as opposed to frame rates, but it should be appreciated that except in very controlled artificial environments extra pre and post processing would undoubtedly be necessary.

8.5. RATES OF TRACK

Given that a target position may be redefined every 40ms a pertinent question is how fast may the target move? If the sampling rate theorem is taken into account then absolute upper bound may be given by:-

$$F = (\text{OUTPUT SAMPLES PER SECOND})/2 \qquad (78)$$

$$F = 25/2 \quad H_Z$$

$$F = 12.5 \quad H_Z$$

True, the image may be found up to this rate (ignoring sensor scan problems) but the rarity of samples makes the output signal very imprecise and of little use. The problem is further compounded by inherent time delay in the system. Because of this the location process always reveals where an object used to be rather than where it currently is. In tracking applications it is of little surprise that the principal emphasis is on pinpointing the current target position, which therefore necessitates a rather tricky operation, namely prediction.

Any physical control system will have some degree of input to output lag. In the experimental set up delays existed within the video capturing, signal processing and result outputting processes. Consider then the graphs of FIG (52), from which it is evident that the magnitude of the unpredicted response is fairly accurate over the specified spectral range but the phase delay increases with frequency. Closer examination of the plots yields an approximate time delay of 100ms.

i.e. $\phi_D = W.T.$ W = ANGULAR TEST FREQ. (79)

= PHASE DELAY (RADS)

then $T = \phi_D / W$ (80)

at POINT A IN FIG (52)

$$T = 1.5x(2x\pi x2.34) = 100mS$$

(as straight line approximation, this is valid for all f considered.)

If the tracking process was motivated by some desire to position a laser designator on a motile target for example then the time delay must be compensated for, to position the spot with accuracy. For the intentions of experiment the notional laser beam was equivalent to the cross wires on the auxiliary nomination display (which has appreciable lag!), although the discussions are relevant to alternative actions such as camera positional control.

8.6 PREDICTION OF POSITION

For a given point in time it is possible to make estimates of target velocity and acceleration based on very recent positional history, e.g.

Given X_2, X_1, X_0

where X_2 is most up to date positional measure and

$$X_1 = z^{-1}X_2 \quad (81)$$

$$X_0 = z^{-2}X_2 \quad (82)$$

approximations to velocities can be obtained as,

$$\dot{X}_2 = \frac{X_2 - X_1}{T_I} \qquad (83)$$

$$\dot{X}_1 = \frac{X_1 - X_0}{T_I} \qquad (84)$$

similarly for acceleration

$$\ddot{X}_2 = \frac{\dot{X}_2 - \dot{X}_1}{T_I} \qquad (85)$$

where T_I is the intersample time period which for frame rate tracking is 40ms i.e.

$$T_I = 40\text{ms}$$

from linear equations of motion

$$S_2' = U_2 T_D + \tfrac{1}{2} A_2 T_D^2 + S_2 \qquad (86)$$

where U = velocity

A = acceleration

T_D = time delay of prediction

S_2 = unpredicted position

S_2' = predicted position

$$S_2' = \dot{X}_2 T_D + \tfrac{1}{2} \ddot{X}_2 T_D^2 + S_2$$

note that in general

$$T_I \neq T_D$$

the equations are usually normalised so that $T_I = 1$ therefore T_D of 100ms = 2.5 units.

Using the above equations it is possible to make predictions of the current target position based on a knowledge of where it used to be. FIG 53 shows plots of relative magnitudes and phase

delay against frequency for a fixed predictor constant T_D = 100ms = 2.5 units. Scrutiny of the results yields the conclusions that the simple general purpose predictor works well at $1H_Z$ and is acceptable up to about 2 - 2.5 Hz. The upper rate is related to the time delay which for 100ms gives a phase shift of $\pi/2$ radians for a sine wave of 2.5Hz. This may be expressed as an empirical "rule of thumb".

Magnitude overshoot and phase equivalence is reasonably acceptable up to the point when the time delay is equal to $\pi/2$ phase shift. Beyond this limit the predictor is forecasting too far ahead to be accurate.

Reducing the time delay or including apriori information about the motional constraints of the target can raise the bandwidth of the predictive process, but in doing so will approach effects caused by sampling problems. Note for 2.5Hz there are 10 samples to represent a cycle, which is approaching the limits of workability.

FIGS (54a-55b) show a simulation of actual and predicted sine waves for various frequencies, clearly demonstrating the degradation with frequency.

The blob tracker used for dynamic measurement is an intensity or binary image region follower. Therefore if an observed scene can be reduced to this form, i.e. target = 1, background = 0 then the process will be valid. As reported in Chap [5] the multispectral

thresholding method is rather good at segmenting colour scenes into target and non-target regions, either directly into binary or via colour mapped binary. Either output will remove the restriction of tracking white objects on dark backgrounds and multimodal images may be considered. PIC (61) shows a moving circular target in amongst a number of distracting coloured objects. (Note there is no significance in the target being white.) In PIC (62) the image has been thresholded from parameters learnt by nominating the required target.

The small rogue object at the bottom of the picture is dominated by the larger principal object due to the area averaging effect so tracking is maintained. Larger distractions are dealt with later. PIC (63) shows the target having dipped behind the red book at the bottom right of the screen. PICS (64-65) demonstrate the effects of nominating other picture areas. (Note that the change in colours is due to exposure differences of the still photographs) Because intensity tracking can be achieved within a field period there is ample time to compute and examine the multiple colour histograms over the designated area, provided it is not too large. Moreover as the double-edged thresholding itself is a hardware utility requiring no processing time, the technique is well within 25Hz operation, even with successive frame colour adaptation.

8.7. PATH FINDER

Under suitable conditions the tracker is a fast an efficient algorithm for finding object bearings. Its averaging nature makes it fairly resilient to noise and enables it to reject clutter thrown up by thresholding providing it has little area with respect to the area of interest. If the application of a process once per frame is successful why not apply it several times? PICS (66-67) display various stages in the use of PATHFINDER. Basically this is a stratagem whereby an area is nominated, thresholded and then four horizontal slices are taken through it. For each a central region co-ordinate is "tracked" and by a similar mechanism two edge searches yield boundary "tracks". The application to images is demonstrated in PICS (68-69) where blue and black folders are nominated.

Now the incentive to implement PATHFINDER was not motivated by a curious interest in lever arch files, nor indeed in the conventional tracking sense of a moving object against a stationery background. In fact PATHFINDER normally assumes a fixed "target" and a mobile camera. These conditions apply to autonomous vehicles, which are currently attracting a great deal of attention (refs. 93,94), one aspect of which being to configure them for self guidance along roads. To do this at an acceptable speed necessitates very high speed computation and so a preliminary investigation was undertaken to assess whether the low level signal processing was within the scope of VOSTTAC1. PIC (70) is a snapshot of a road captured from a RARDE test video.

The segmentation procedure was to maintain a fixed position learning window positioned on the road, from which to measure the peak colour combinations of the surface. The results of single and double peaked multispectral thresholds are recorded in PICS (71,72). Note that the double peaked method if used carefully can yield less "perforated" regions than the single variant.

It should also be appreciated that the output displays are in mapped binary form, which gives an acceptable visual appearance but the ≥ 0 and $=0$ tests apply. Under such conditions it is admissible to use the PATHFINDER to attempt to find the centre and edges of the road. PICS (73 - 76) show the procedure engaged on a "real" road video. Owing to the area averaging nature of the tracking "slices" as the vehicle begins to veer off course, the centre of road area shifts from its normal position and produces an error signal. This is indicated by a broken line pointing back to the desired direction, which could be used as a means of correction, thus keeping the vehicle within the confines of the road. Perhaps a semi-autonomous mode might require a driver to guide the vehicle based on the error trajectories alone.

Whilst the pathfinding method is acceptable under certain conditions and fairly robust to noise, it would be a very brave (or foolhardy!) driver who would use the system without taking stock of the following implications.

a) Road and background must be of contrasting colour combinations. This is quite often the case, but cannot be guaranteed and the condition fails under low light conditions. Increases in contrast sensitivity can only be made at the expense of tolerance and noise immunity.

b) Double peak multispectral thresholds are likely to yield more solid regions than single ones, but only when there are two or more dominant peaks within the histograms. Failure to satisfy this qualification can result in completely erroneous colour attributes, if secondary so called "peaks" are in fact just noise.

c) Relearning of the surface colours must occur as rapidly as practicable, i.e. frame rate. This is imperative to cope with varying illumination as well as gradual surface changes. There must therefore be a very high confidence/probability that the adaptation window is actually positioned on the road. This conviction would be gained as a result of some previous processes and without which the machine could quite easily adapt itself to roads, rivers, pavements, passing dogs, etc!

8.8. MOVEMENT DETECTION

Multispectral thresholding techniques can be powerful segmenters but so too is an attribute that has yet to be mentioned, namely movement. This can be a target extraction scheme of remarkable performance, particularly with regard to its ease of extraction.

If a "still life" picture is captured within a frame store it is likely to be identical to successive images if quantization and noise effects are ignored. Therefore if the stored image is subtracted from a later version of itself, the result should be null, i.e. blank screen. If however there is a local discrepancy between the two pictures, their subtraction will show up as a disturbance, i.e. bright area. Such effects can be produced by moving targets.

Consider PIC (77) which is a view of an uncontrolled or "real world" multi colour background of cabinets, boxes, etc. If this were directly subtracted the output would be blank. In PIC (78) a disturbance is introduced in the form of a man (with a pipe!) walking past the cabinets which is clearly visible in movement space. The transform can be used as an alternative means of initiating tracking as opposed to thresholding or edge enhancements.

For example it is very useful for remote surveillance where a stationery camera is perhaps waiting for intruders. One point to remember is that should the observed object come to a halt then it will vanish from display so there should be a last position hold on any tracking activity. Perhaps the most impressive strength of movement detection is its ability to reject very complex backgrounds. PIC (79) is a prime example of a cluttered scene,

displaying an analog computer with its usual associated mass of wires, etc. In addition there is a moving black pendulum of irregular shape, swinging near the top of the picture. The raw image does not make this at all obvious, yet the movement space of PIC (80) is dominated by the emphasised pendulum.

8.9. DTP BACKGROUND SUBTRACTOR

VOSTTAC1 can background subtract with consummate ease and at a rate suitable for on-line displays. The function is a built in primitive of the DTPS. To use this the first step is to ensure that a frame store contains an image and that the read before write store access option is selected. As the next image arrives via the video scanner it is compared on a pixel by pixel basis with the stored image using DTP firmware and the magnitude of the difference is then stored and displayed. Movement works on all 3 colour spaces so there is choice of confidence functions, i.e. OR, AND etc. The function requires absolutely no slave processing time whatsoever, but as two frames are necessary for each transform the minimum cycle time is limited to 80ms. This is still quite acceptable for visual real time displays and it must be emphasised that all this period is available for any required processing. The DTP subtractor function may be considered as a separate parallel processor. To calculate the software equivalence of this firmware assume that to subtract two values requires 3 cycles, i.e. LOAD, SUBTRACT, STORE. This would have to be performed for every pixel, thus making the total number of instructions:

$$T_t = \text{SCREENSIZE} \times 3 \qquad (87)$$

$$= 577 \times 360 \times 3$$

$$= 623{,}160 \quad \text{cycles}$$

Based on a (5MIP) TMS32010 this is equivalent to a processing time of:

$$P_t = 623{,}160 \times T \qquad (\text{where } T = 200\text{x}10^{-9}\text{s}) \quad (88)$$

$$= 623{,}160 \times 200\text{x}10^{-9}$$

$$= 125\text{ms}$$

So even with a naively low estimate of the number of instructions per pixel, over 3 frames worth of TMS32010 processing would be required. The inclusion of the modest hardware of the DTP movement detector would therefore seem something of a bargain.

Aside from spotting movement attributes, picture subtraction can also reveal missing objects from otherwise identical images. For this reason it may be employed within inspection systems, flagging errors if say a component is missing from a circuit board.

Another area in which movement might be a useful attribute is where the camera itself is moving, perhaps when mounted on a vehicle. this type of application is by no means as straightforward as those mentioned previously. If pixel by pixel subtraction is applied directly, a great deal of output will be produced as the vehicle movement shifts and distorts the comparative images. As a reference point one might propose that

the most distant visible point will move the least and be undetected. The horizon obviously belongs to the set, but generally it fails to vanish and furthermore is often the brightest output of the entire image.

This apparent contradiction is due to the fact that the horizon usually is of a high contrast boundary which shifts its relative position as the vehicle travels over bumps and slopes and hence generates a high subtracted output between successive frames. This typical pitfall suggests that much more research is required in this area, primarily directed towards high speed alignment of equivalent pixels in different images. Whatever detailed conclusions are eventually reached, it is unlikely that the simple subtractor used for surveillance be adequate for autonomous vehicle usage.

8.10 MULTIPLE TARGETS

No matter what primary segmentation/extraction technique is employed, there is always the likelihood of detecting more than one region of interest. Recall that in Chapter [5] a multiregion segmentation scheme was discussed from the point of view of rejecting unwanted objects and noise. However it may well be that several objects need to be tracked simultaneously. Whilst a full discussion of multitarget tracking is beyond the scope of this

work, some investigative research has been undertaken, basically extending the blob segmenter. Although the associated co-ordinate grid is coarse (but can be improved), the segmenter does perform all the necessary early processes of a multitarget tracker, i.e. finds all distinct blobs subject to a minimum distance criteria and lists their approximate areas and central co-ordinates. All this at frame rate. Any additional tests could then home in on these blobs and test for attributes such as shape, colour area, etc. If say area is used as a criteria, the blob list may be sorted into order of ascending size. Therefore particular tracking channels would point to specific entries e.g. channel 1 tracks largest, channel 2 the second largest, etc.
PICS (81-82) show multitarget tracking and although only one target is shown per picture due to only having the single cross indicator, all blobs were actually tracked simultaneously. To change the cross display assignment from one target to another merely required altering the blob list pointer.

The problems associated with long term tracking are really the responsibility of some high level monitoring process based on past trajectories. As mentioned previously this had not been a primary goal of the research effort, but discussion of a simple decision making process will at least highlight the major problem areas.

Given that N targets are visible and separate, associate them with N windows of N tracking channels, where a window might be any size up to full screen. If the number of targets at the next track cycle is different to N then one of several events have occurred.

a) TARGET PASSED FROM VIEW (DEPARTURE)

b) NEW TARGET IN VIEW (ARRIVAL)

c) OVERLAPPING TARGETS (OCCLUSION)

Case a) should normally be known in advance, based on prior target positions. If it occurs unexpectedly then some maximum probability criteria could be used to isolate the absentees track channel. Case b) is almost invariably unannounced and requires that the blob list be incremented and a tracking channel allocated. In addition, the criteria by which channels are associated to particular blobs must be reviewed with regard to loci trajectories, so that the new target does not swap channels with existing traces. Case (c) occurs initially when two objects are separated by a distance sufficiently small to be beyond the resolving power of the segmenter. Effectively two blobs will merge into a larger one. The two intersecting objects should be identifiable from their trace histories and their channels could be both assigned to the composite blob. This redundant channel

tracking would continue until such times as the primary targets are again observed as separable, or until they failed to be of interest, e.g. out of view, timeout, etc.

Although the analysis of tracking paths may involve complex decision making, drifting into the realms of Artificial Intelligence systems, the data worked upon is high level, i.e. co-ordinates shape area etc. Therefore the processes are not nearly as slow as the actual blob level tracking which although low level acts on very many pixels. It is therefore the performance of the rudimentary tracking process which limits the system cycle time. Speed considerations for a particular example applied to the segmenter where given in Chapter [5] but for a more comprehensive performance evaluation what is required is the speed variation with respect to variables such as the number of targets and picture "activity".

The multitarget tracking algorithm can be lumped into five functional blocks:

1) WINDOW DUMP

2) BLOCK AVERAGE (4 PIXEL SUMMATION, FOLLOWED BY 16 PIXEL SUMMATION)

3) PRIMARY GROUP ASSIGNMENT

4) SECONDARY GROUP ASSIGNMENT

5) AREA, XY CALCULATIONS AND TERTIARY GROUP ASSIGNMENT

Stage 1 is easily dealt with as a $\leq$ 1ms constant. The averaging process is fairly regular but the overall processing time depends on the rank of the window considered, i.e.

$$T_2 = T\sum 4PIX + T\sum 16PIX \qquad \text{CYCLES} \qquad (89)$$

where $$T\sum 4PIX = \frac{RANK^2 x 27}{4} \qquad (90)$$

$$T\sum 16\ PIX = \left(\frac{RANK}{4}\right)^2 x 39 \qquad (91)$$

$$T_2 = \frac{RANK^2 x\ (27+39)}{4 \qquad 4} \qquad (92)$$

$$= \frac{RANK^2 x(36.75)}{4}$$

$$= 9.19x(RANK)^2$$

As the rank increases so does the number of blocks to be further processed. The primary group assignment is therefore also dependent on the rank, but in addition it is affected by the ratio of active to inactive blocks.

$$T_3 = Nx20+Mx16 \quad \text{CYCLES} \qquad (93)$$

where N = NUMBER OF ACTIVE PIXEL BLOCKS

m = NUMBER OF PIXEL BLOCKS

which can be written

$$m = \left(\frac{RANK}{4}\right)^2 \qquad (94)$$

therefore

$$T_3 = Nx20 + \left(\frac{RANK}{4}\right)^2 x16 \quad \text{CYCLES} \qquad (95)$$

Primary group assignment is therefore dependent not only on characteristics of the test, but also the type of input image. The process produces a list with one entry for each active block which is then used by the following stage. The secondary assignment can therefore be measured in terms of this list size. i.e.

$$T_4 = \text{LIST SIZE} \times (50 \times \text{N°TARGETS} + 12 \times \text{LIST SIZE}) \qquad (96)$$

The secondary assignment identifies the number of distinct target groupings and therefore a new variable has entered into the timings which again is dependent on the particular test image.

Stage 5 is the most difficult to analyse depending on the number of targets, list count, and rank, i.e.

$$T_5 = N°TARGETS \times (75 + LIST\ SIZE \times (19 + \frac{45}{N°TARGETS}) + RANK^2 \frac{13}{32}) \quad (97)$$

Performance graphs for the various stages are shown in FIGS (56 - 60). With the number of variables invovled it is difficult to make sweeping generalisations about the speed of tracking, but experimentally 6 targets of a randomly chosen image were tracked within frame rate. For more specific conditions the relevant portions of the characteristic curves should be consulted.

Clearly if more precise co-ordinate information is ever required a higher resolution segmentation would be necessitated. Nevertheless it would be inadvisable to do this as an initial step. It would be more appropriate to refine the test successively about the coarsely defined blob centres in an ascending pyramid type approach. In this way the amount of redundant data accessed is kept to a minimum. For example if the precursory pass yields a number of coarse blobs, then higher resolution windows could be positioned about each and the process repeated. This process could continue until either full resolution is reached or sufficient selectivity attained. Obviously the more sophisticated the classifications the greater the processing "load", but it should be noted that the timings for

multitarget and tracking given in this chapter relate to a single slave sequence and so there is considerable scope for increased computational power through parallelism.

By now the image analysis engine VOSTTAC1 has been well and truly introduced and shown to be capable of performing in the major areas of high rate picture processing. Its speed is due to a careful design philosophy, rather than consolidation of a fixed algorithm and therefore it is sufficiently flexible to tackle a whole host of practical problems, whether they be tracking, inspection, etc. To begin to appreciate the vast scope of applications a representative review is presented in the following chapter.

CHAPTER 9

APPLICATIONS

9.1 USES FOR VOSTTAC1

This chapter is concerned with some practical application problems which one might endeavour to resolve using VOSTTAC1. Most of the major processes have already been investigated in previous chapters and therefore applications can be dealt with under the established topics of enhancement/correction, shape detection, tracking, colour and edge detection.

9.2 ENHANCEMENT/CORRECTION

There are a number of simple picture modifications which can make almost magical improvements to visual appearance i.e. as seen by the human eye. Many of these are used to increase image contrast by making better use of the available digitisation levels. These are known collectively as point operators, which are characterised as a mapping from a single input to single output pixel with a grey or colour level change. Often all that is required is the add, subtract and multiply functions which are readily implemented at high speed with modest amounts of hardware. The operators can be even more valuable when applied locally as opposed to globally, perhaps to compensate for photometric non-linearities. An example of a very complex point operator was used on the Mariner/Venus/Mercury mission 1973 (ref. 25) for which a nine point piecewise linear digitisation characteristic was defined

for every observed pixel. To a lesser extent a modification to the digitiser characteristic is often required when copying photographic films. The response of a still film to illumination is governed by a Hurter and Driffield curve, a typical example being shown in FIG (9) with density against the log of exposure. To ensure that the scanner responds proportionally to the actual optical density means that this curve must be compensated for.Usually this is achieved by passing the signal through a function producing the negative of the transmitted logarithm. Practically this can be realised as a digital filter but is often implemented in analog form.

A problem more associated with aquatic activities appertains to the improvement of underwater images. More specifically related to deep sea applications such as inspection and mine clearance, where low contrast and noisey picture are the norm. Beneath the waves there is little ambient illumination, certainly around the coasts of Great Britain and so image intensified cameras must be used. These are characterised by high sensitivity, but equally so by a high noise factor. Increasing the illumination artificially is not generally successful as the water contains silt in suspension which reflects back the rays of light like a mirror. The classic way of improving the signal to noise ratio of any image is to average several over a period of time. Whilst VOSTTAC1 has not been demonstrated in this role, it is quite capable of filling it with only minor modifications. Recall that a sub-section of the DTP can perform background subtraction using

programmed logic. That same physical circuitry with only firmware changes can produce automatic averaging, i.e. each pixel is compared with its equivalent in the next successive frame and the average stored. In keeping with the properties of the movement detector the averaging would not require any slave processing. Also if the input sensor is significantly non stationary, pixel alignment must be more carefully considered.

Infra red television systems are used extensively for tracking and surveillance. The popularity stems from their 24 hour capability, i.e. they can work at night, detecting missile heat paths, hot engines, and people. A peculiarity of many IR sensors is 'streaking' along horizontal scan lines. This is a non input dependant change in average signal which varies with vertical co-ordinate position. A simple compensation which effectively removes the mean streak (no pun intended!) is sometimes known as grey level detrending (ref. 84).The process calculates and adapts a mean value along a line, by using some recursive filter, and for every line point the latest mean is subtracted from the pixel value so that only the variation is stored.

Although all the experimental work tackled using VOSTTAC1 has been based on the visual spectrum there is no reason why IR cameras could not be used, providing they are, or can be made to appear P.A.L. compatible. Infact provided digital data can be forced into the frame stores, VOSTTAC1 can attempt to analyse it irrespective of primary sensor type.

Aside from transformations to grey level another aspect of image alteration involves geometric image correction. Some sensors by their nature do not yield captured images in their usual recognisable form but subject them to regular spatical distortion. Satellites are prime offenders, their relayed pictures perhaps depending on the curvature of a planets surface, its distance from the craft and the angle subtended to it. Ref. 25 gives examples of this type of problem including ones associated with the VIKING LANDER (Martian probe). The related rectification process, required mapping of a square but distorted image to a sector like array with correct spatial proportions.

VOSTTAC1 is unlikely to be able to effect geometric corrections at very high speed (frame rate) due to the difficulty in calculating addresses for a very large number of input pixels. However as this kind of rectification is almost invariably an off-line activity, the speed penalties are not severe.

9.3. SHAPE INSPECTION

Shape is a powerful criteria in many decision making processes. A common robotics problem is to pick up a particular object at variable range and random orientation. The shape classification capabilities of VOSTTAC1 would seem well suited to this type of

application being independant of object size and orientation. The technique has been validated for 2-D shapes with relearning but further investigations may be needed to ensure robust 3-D model approximations. This is necessary to tackle the problems caused by occlusion of multiple objects. Although this is a hard nut to crack, probably requiring a research study all to itself, it would seem achievable and an area where VOSTTAC1 perhaps has an edge over other high speed systems.

With the growing interest in letting loose autonomous and semi-autonomous vehicles on roadways a number of secondary applications for shape detection become apparent. Skipping over the extremely complex problem of maintaining a vehicle moving along a road there are a number of very distinct shapes that would pass through the field of view of the moving camera, namely traffic signs. By design these are strikingly noticeable to the observer in terms of shape, colour and position relative to the road. For example, given a rough area of interest it would be a simple matter for the vectorscanning shape classifier to differentiate say between triangular warning signs and circular prohibitory signs. This could be followed by colour thresholding techniques to further define the category of signal. These actions could be used to make a human driver more aware of likely hazards, perhaps in the form of an artificial verbal remonstration (i.e. "You are suffering from motorway madness, please slow down"!)

Alternatively the information might form part of a route and speed logging exercise, akin to the black box flight recordings.

For fully autonomous vehicles there is likely to be the situation when they get hopelessly lost, perhaps due to a failure of some geographical satellite based information link. In which case it is not unreasonable to suggest that the steps taken to resume their mission would be similar to those taken by a human driver in the same predicament. Assuming that the machine is insufficiently intelligent to ask a policeman and even if it were it would know that there is a very low probability of finding one when needed, it would probably look for road signs. This necessitates an extension to the symbolic traffic sign detection to enable it to read directional placards, which demands a degree of skill as a text reader.

Preliminary investigations have been conducted using VOSTTAC1 to analyse textual information using a coding rather than a template matching procedure (ref.95). This generated some encouraging results, but also isolated problems caused by low contrast images and variable fonts when applied generally. Fortunately road signs use bold and regular lettering which suggests that there interpretation would be an attainable goal.

An autonomous road following vehicle could not drive all the way from here in England to Australia. This statement is a pathetic attempt to try and link the previous application with one involving kiwi fruit. As western palates become more adventurous and cosmopolitan, this fruit is developing into a growing source of income for the Australian economy. In years gone by there has been criticism of the calibre of the produce after exportation, which is a contributing factor to the quality consciousness of distributors.

This has reached the extent of using image processing to monitor the health of the product. One aspect of this which is currently under investigation (ref. 96) is the detection of irregularities in the shape which should be a smooth ellipsoid. Tests are normally carried out on a 2-D silhouette. Although the shape coder of VOSTTAC1 is currently restricted to measuring deviations from circular boundaries it would not seem an impossible step to normalise the measurements with respect to an ellipse. This could then be a very useful technique in the control of kiwi fruit because of its high speed. More generally this type of action is typical of many inspection problems, where some departure from the norm is anticipated.

9.4 TRACKING ROLES

Classically tracking is a military concern, monitoring the positions and paths of aircraft, ships, tanks and missiles. Staying within this vein it might be feasible to use VOSTTAC1 as a passive loci tracer, either to assist eventual human interpretation or to take some semi-autonomous action. Visual and IR tracking are relatively short range by comparison with conventional radar, but on the plus side the passive systems can work very quickly and precisely.

The Falklands war made it tragically apparent that there was a need to counter sea-skimming missiles such as exocet after primary defences had been flanked. One solution is to blast the approaching projectile/target with a very high rate of fire machine gun effectively throwing a metal wall in its path. If the target was nominated by an operator VOSTTAC1 could well be used to not only track it, but control a gun platform to bear on it.Providing the rig could be made physically and algorithmically robust for the given environment, it would be a compact and inexpensive short range defence against missiles and aircraft for say small ships. Of course, this would not be the be all and end all of defence systems, but it could make the difference between oblivion and survival when all else has failed. After all cheap and short range weapon systems have saved the day throughout history, for example Goliath would not have let David keep his good looks if not for the slingshot!

Keeping on the theme of monitoring and defence an application for which VOSTTAC1 is ideally suited is that of remote sentry. This involves the strategic positioning of a visual or IR camera as a sensor of undesirable events such as human or vehicular intrusion. VOSTTAC1 could easily movement detect or trace heat paths to either produce a warning or effect some countermeasure using its control capability. In commercial security systems for instance, it is quite common for a guard to sit in front of a bank of TV screens to detect any criminal activity. In all probability for 99.9% of the chap's career the displays will be uneventful and when something actually does happen he will likely disbelieve it or be asleep. Automatic systems have the advantage that their attentiveness does not wane.

Moving away from defence and security, the production of guidance signals to force a vehicle to drive along a road is of interest. The complexity of the problem when applied to general road scenes is beyond the scope of VOSTTAC1 by such a margin that the universal application of road following is unlikely to be within the capabilities of technology and human endeavour for some time to come. Certainly there are conditions under which road followers function acceptably as in the case with PATHFINDER.However this imposes constraints on environmental variables which in practise are usually beyond control. Qualifying tests with precise conditions is falling into the

"artificial" trap, e.g. if it is only possible to work in strictly controlled environments then why not make them visually obvious by say, painting a fluorescent pink polka dot line down the centre of the road.

Although at present totally autonomous vehicles would seem inapplicable, semi-autonomy would seem a desirable and achievable ambition. One typical application is the convoy or follow my leader vehicle train. The motivation being that instead of having one driver per vehicle, the majority would be computer controlled "drones" following a single human driven leader vehicle. Under these conditions tracking is much simplified only requiring identification of the vehicle in front either by the true image or via some light or marker code. As an extension local road following could be applied to assist correct cornering and obstacle avoidance. It must be mentioned that an important system function would be to detect when the convoy commander goes haywire, more technically described as an erroneous departure from the recommended or safe course. Without this precaution the vehicle train might be better described as follow my lemming!.

On the entertainment front tracking is of interest to television companies. There is a problem in fast moving sporting events in keeping the camera up with the action. Down hill skiing is a typical example where sportspersons are moving very quickly indeed and there is the possibility that the cameraman can not react quickly enough to maintain them within the field of view.

This situation is taken as obvious candidate for an artificial tracking system. There is a similar problem in football matches where the ball must be kept in view. The definition of algorithms to disseminate the ball from the tangle of arms and legs of the players is a more difficult task but may be reachable especially if the ball is made more noticeable to a particular sensor type.

9.5. COLOUR TESTING

Colour is a very useful yet underused image segmentation and identification attribute. The lack of exploitation stems chiefly from the increased cost relative to luminance only systems. With the plumetting prices of colour cameras, monitors and memories this penalty is unlikely to remain.

In artificially controlled environments such as factories chrominance is a simple means of differentiating between disimilar objects. In theory VOSTTAC1 can handle up to 4096 different colour combined objects. If some colour bar coding was employed this number could be extended immeasurably. The colours need not be falsely imposed, advantage being taken of those naturally occuring. For example, hue may assist putting the white paper into a light brown box etc. The system could easily be modified to different types of applications using the nominated learning facility provided. In this way the equipment could be adapted to any situation where different colours imply disjoint regions e.g. sea and beach, runway and grass, fields and roads.

The use of multispectral thresholding strategies to perform target and road tracking has already been well covered, but another colour intensive application which has not been mentioned concerns aircraft taxing. In 24 hour all weather airports it is not always possible for pilots to directly observe features of the ground stations and therefore taxing is just following a path of coloured lights or beacons to a particular bay. VOSTTAC1 could direct such a task with relative ease. Furthermore if the lights were controlled more intelligently they could be used to intimate more complex modes of operation. Similarly baggage cars, and airport buses are all candidates for simply directed semi-autonomy.

9.6. APPLIED EDGE DETECTION

This technique is used in a growing number of image analysis applications. Even though the method discards regional information, it can define their boundary positions fairly precisely. By comparison, a pure thresholding approach will yield poorly defined border locations due to sensitivity to the threshold level. In fact under some circumstances thresholding is a totally inapplicable means of segmentation. For example there is little likelihood that the snowtracks of PIC (34). could be reliably extracted using a level based procedure, due to the low contrast between the tracks and snow. However the convolution with first or second order differential operators makes the trail

very conspicuous. If this locally enhanced picture information is then coded mathematically via some more global description (perhaps using the Hough method) then the edges can be regarded as detected. The general strategies is applicable to many different types of image, e.g. aerial photographs, underwater images etc. Whilst image enhancement alone is often sufficient for subsequent human interpretation, autonomous action may only procede by detected or encoded images. The production of the high level picture description may also be considered as a data compressional effect, which can be of use when storing sub-images and models etc. For example the trivial case of a single line against a uniform background only a few words of description would be needed to encode it, as opposed to the 1/4 MEGAWORDS required to store the raw image.

9.7. CASE STUDY

Various investigations using VOSTTAC1 have already been discussed in earlier chapters, one which has not is a factory inspection system, that has been the subject of a precursory investigation.

A manufacturer produces a great many aluminium slugs every day, a proportion of which are faulty. It is currently a manual process to remove defective items, which understandability is a tedious and time consuming task. The only other alternative to this at the moment is to sell unmonitored batches at lower prices.

Therefore a means of automatic passing or failing slugs is desirable. PICS (83, 85, 87, 89, 91, 93) show various fault conditions for small slugs which can be described thus;

a) SIDE SPLIT (i.e. Top, Bottom, Right, Left Etc.)

b) INDENTATIONS/PIPS (Small dents and surface discontinuity)

c) LINE CRACK (Fissure chord of circle)

d) FAULT LINE (Surface line as distinct from Crack)

e) CUT (Niche does not extend to perimeter)

f) NORMAL

As these effects produce surface discontinuities it is not unreasonable to expect that they would be highlighted by local differential operators, such as the Sobel or Prewitt mentioned earlier. A Prewitt was applied to fault images and the results are shown in PICS (84, 86, 88, 90, 92, 94) The imperfections became patently obvious which was remarkable as only the standard utility version of the edge enhancer was used without modification, i.e. fixed threshold and no contrast enhancement. There is then, obvious scope for improvement to yield even better results by making the transform more dependent on the ambient conditions of test. As far as pass/fail decision making is concerned one simple method would be to first strip off the outer circular boundary by either a searching process or intuitively by assuming a known position for the central slug area. It is then a simple matter to find a figure of merit for the object based on the integration over the surface of the edge information. If this

value is greater than a permitted threshold obtained by training with acceptable "pass" slugs, then the item should be rejected. This is a very simple and quick means of inspection and should be cheaply implemented. A refinement on the pass fail system might be to identify the class of fault to perhaps produce some statistical performance measures of the process to identify shortfalling in production. The lines and curves could be handled using the Hough transform mentioned in Chapter 6. Pipping is probably best handled by a template process which emphasises "spots" with indentations (surface noise) being a failure of either classifier but with a large amount of edge clutter. All these processes have been well covered thus far and are readily implemented for high speed operation. The method is therefore practicable and likely to be applicable to other inspection problems, e.g. larger discs are treated in PICS (95 - 98).

There is another kind of fault worth mentioning whilst on the subject of aluminium disc inspection, which could be analysed in the future if sufficient interest. Aluminium being of fairly soft material is likely to dent if treated roughly, so discs occasionally have flats on their otherwise circular perimeters. Recall that the vector scanning shape descriptor of VOSTTAC works directly on boundary deviation from a circle. Therefore if sufficient sensitivity, the test is a ready made dent dector for variably sized and orientated disc slugs yielding the number of dents and their cummulative energy.

Hopefully the foregoing discussions have illustrated the diversity of duties which VOSTTAC1 might realistically perform. Nevertheless the system most suitable for investigative research is unlikely to be identical to a production unit. Therefore a number of proposals for future development have been put forward and expanded in the following chapter.

CHAPTER 10

FUTURE DEVELOPMENTS

10.1 TOWARDS VOSTTAC2

Of all the successful electronically engineered products on the world market today there must be few that are exact copies of a MKI prototype. The architypical "system" is by and large designed for research experiments and knowledge acquisition within the intended extent of application areas. Therefore when applied to specific practical problems the original machine is likely to suffer from both over and under engineering. It is then essential to identify areas where modifications can be made and appreciate their associated improvements and degradations. Fortunately VOSTTAC1 was defined with a view to technology upgrading being based on a modular framework.

Before deliberating on VOSTTAC1's successors, it is necessary to question the motivation behind changes, i.e. what are the goals of a production unit? Some relevant criteria are listed below:

1. ACCEPTABLE PERFORMANCE
2. LOWEST COST
3. SMALLEST PHYSICAL SIZE
4. LOWEST POWER CONSUMPTION
5. MULTI-PURPOSE
6. EXPANDABLE
7. SIMPLEST TO BUILD

8. PHYSICALLY ROBUST

With these points in mind the major elements of VOSTTAC1 will be reviewed.

10.2 DIGITAL VIDEO SYSTEM FACTORS

When the original system was being specififed it was stressed that as a research tool the machine should operate as fast as possible within the limitations of the approach and technology. For the VOS machine to work at the very high speeds specified, the addressable memory needed to provide equally fast random access. For this reason the frame stores were implemented using large static rams. These then introduced a significant cost and size increase for the machine. Whilst this is necessary (and more affordable) for highly specified systems such as in MOD usage, for a very great many applications the price tag would prove prohibitive. In such cases the decreased cost, size and power consumption of modern dynamic rams would be sufficient reason for derating the transfer speed. On the assumption that a large proportion of the "market" could tolerate reduced transfer rates it would be possible to produce compact and relatively inexpensive frame stores from hybrid dynamic ram modules. Furthermore it would be conceivable to store more bits per colour as the additional size and cost penalties would be less severe than for a static ram store.

Even with the falling costs of memories, it is inefficient and ultimately less competitive to use more than is necessary for a given purpose, and so both store size and depth reductions must be considered. Variably sized stores pose no problems as VOSTTAC1 can vary the number of samples per line in order to fit a given memory capacity. Depth of store is more easily varied and should be the minimum sufferable by the most demanding process under the given conditions.

The benefits from reducing the number of quantization levels of a sampled signal based on other than economic grounds are not immediately obvious, e.g. if a 16-bit processor, why not have 8-bit plus arguments. Indeed for software solutions a large number range is acceptable, but for the special firmware usually employed to speed operation, there are great penalties associated with increased word length. For example if some complex function is required on two 4-bit arguments a 256x4 look up table could be used in the form of a readily available 8-address input prom. If on the other hand two 8-bit arguments are assumed a 64K x 8(16 address inputs) look up table would be required which is too large to be practical (proms the size of an entire Z80 address space).

The ideal video storage system would allow trade offs between sample window size, resolution, quantizing levels and number of pictures stored. Proposing this practically should not be considered overly ambitious as the key points have already cropped

up during the design of VOSTTAC1 albeit in different parts of the system. If special care is taken in reducing the amount of picture memory then it is essential that these stores are exploited to full capability. It would therefore seem wise to provide a flexible front end conditioning unit to effect data control and possibly level transformations.

In the current "decadent" system 12 bits of storage are provided per pixel, nominally configured as 4-bits for each primary colour, e.g. RGB, although all are sampled and buffered to 7 bits. Acquiring increased storage allocations for the specific colours or luminance is possible with manual intervention but is not a straightforward procedure. Much better would be a software controlled flexible multiplexing arrangement which would allow variable allocation of bits per signal.Level mapping of input sampled data is not always a good thing as it irretrievably affects store data for a whole cycle period. Nevertheless it does have merit when dynamic ranges need to be compacted as might be the case in a reduced memory system. For this reason an input mapping table should be available as an option.

At the very beginning of the video chain, the colour and luminance signals are in analog form and hence can be altered by continuous transfer functions as belonging to simple filters and amplifiers. Yet in many systems other than image processors, analog function generators and filters have been superseded by digital equivalents

due to drift and accuracy considerations. However the bandwidth of television signals is sufficiently large to still favour some analog circuits and moreover they can often entail much lower costs and fewer components than a digital equivalent. Therefore it would seem advisable to provision space for a number of useful circuits including logarithmic film compensator correction, an aperture shading corrector, detrending filter and general 1-D filters.

As far as a production unit is concerned the digital video should be based on a minimal or skeleton system, to add to which the required memory and auxilliary modules for specific task areas.

10.3. MULTIPROCESSING SYSTEM CONSIDERATIONS

The parallel processing system used in VOSTTAC1 used TMS32010S mainly due to their impressive 5-MIPS with special convolution/filtering instructions. However it was emphasised from the outset that the system was not dedicated to one specific type of processor, but provided an environment within which various microprocessors and/or hardware could function effectively. Neither does the system have to use a single type of processor. Providing simple protocols are observed it is possible to mix different computational/functional units, as their bus peculiarities are confined to single board level.

With TMS32010S retailing about the £50 mark there is an understandable temptation to use cheaper general purpose processors, e.g. Z80, 6502 etc. Of course this would dramatically reduce performance analogous to say putting an underpowered engine in a sports car. Whilst technically such a severe derating would be unforgiveable it could make the system more economially attractive to less serious users. For that reason the options of cheap more inferior processors must be supported. A processor which can definitely not be described as either cheap or second rate is the INMOS Transputer (ref. 97) where £500 will buy the top of the range device. During the research project enquiries were made and conferences attended on the then "proposed transputer" which actually took another two years before becoming available. (An interesting phenomena was how it was claimed to be imminently available for release, throughout this 2 year period!). What became apparent was that the Transputer was an elegant and radical new computing unit which demands that it be reviewed here if only briefly.

The key features of the T414 transputer (top of range) are as follows:

* 32 Bit
* 10 MIPS
* 4 Gigabyte Linear Address Space
* 32 Bit Wide 25 Mbyte/Sec Memory Interface

* On Chip Memory Controller
* 2K Bytes on Chip Ram
* 4 Inter Transputer Links
* Advanced 1.5 Micron CMOS technology
* 20W Power Dissipation (500mW)
* OCCAM Programming Language

As a computational building block the transputer is a powerful device, allowing algorithms developed on minimal systems to be accelerated by adding more transputers and making only minor modifications to the occam program. However such systems are not tailor made for real time low level signal processing where data transference and memory access times become a major part of the overheads.

If frame rate image processing is attempted it is imperative that the natural flow of the enormous amounts of picture data be preserved. Failure to ensure this would reduce any system to an off-line analysis tool irrespective of the amount of computational "clout" brought to bear on the captured image. The 2-D arrays of transputers suggested for picture processing cannot support the I/O of image data through themselves as they neither have the memory capability to hold it (4K per device) nor the serial link speed to achieve it at frame rate. Therefore transputer arrays must have intelligently managed external frame stores as might be

provided by VOSTTAC1. To justify not only the cost of the transputer array but the special hardware environment as well is not easy.

For the level of processing considered a great deal of data requires elementary analysis or transformation which is an area where SIMD arrays would outperform transputers. It should also be appreciated that modest amounts of dedicated hardware could (and do in the DTP) outperform either approach for a fraction of the cost.

Transputers appear more suited to fast yet off-line image analysis drifting towards artificially intelligent systems where their considerable power can be unleashed constructively. An example being the new fingerprint classification system built around transputers and commissioned by the Home Office. While governments need not baulk at the price tag it would likely cause widespread apoplexy amongst many potential buyers of image analysis equipment. However should funds become available it would be an interesting exercise to interface a transputer to VOSTTAC1.

The most logical system upgrade would be to include the latest marvel from Texas Instruments the TMS32020 (ref. 98). This natural successor to the TMS32010 at first glance seems just a rehash of an old design, but closer inspection reveals a much more powerful processor. The increased capability is not the result of

some fundamental redesign, but rather removal of many of the limitations which beset its predecessors. Some of the more notable improvements are listed below:

* 544 Words of Internal Datamemory
* 64K External Directly Addressable Program Memory Space
* 64K External Directly Addressable Data Memory Space
* 16 I/O Channels
* Global Memory Interface
* Floating Point Instructions
* Block Moves 80 Mbytes/Sec
* Repeat Instruction - (Makes better use of pipelining than Branch)
* 5 Auxilliary Registers with Auto Increment/Decrement
* Auxilliary Register ALU
* Serial Port
* Wait States on Memory Access
* On Chip Timer
* 3 External Maskable Ints
* Output Flag Pin

Certainly as far as the master processor is concerned an upgrade to a TMS32020 system would seem desirable for its external address capabilities alone. Furthermore the new facilities en masse would greatly assist the system control and executive capabilities making for a smaller but more powerful supervisor. With TMS32010

programs being upward compatible with its successor it might seem strange that the current master has not been ousted from power. There are two good reasons why this has not happened. Firstly the new chip is several times more expensive than its predecessor and secondly but more importantly all the processor development equipment and facilities purchased for the TMS32010 are totally inadequate for the TMS32020. These would necessitate reinvestment and hence significantly raise the considerable cost of the system as a whole. The moral that should be observed is that yes, it is a good idea to update the system when new technology becomes available, but only when a potential customer or user puts forward a convincing argument, e.g. waves a lot of money around!

For instance, a simple LAN between processor boards would provide a fast and convenient low traffic communications link. This would provide more system flexibility, but as its not absolutely essential is unlikely to be implemented without very good reason.

Getting away from the type of modules which actually perform the processing, e.g. TMS32010 hardware etc., the architectural expansion can be considered.
Although the multiprocessor tree arrangement can be theoretically expanded both sideways and downwards ad infinitum , it is unlikely that the system would extend beyond 3 layers due to increased connections and operating system overhead. Moreover the envisaged arrangement would be on 2 levels with 2+ slave boards.

10.4. VOS ENHANCEMENTS

The Vector Orthogonal Scan transfer tool is the cornerstone of the whole system. Physically it extends over two circuit boards and incorporates the frame store addresser, the window addresser, and the state machine development unit, all of which are implemented with a large proportion of high speed logic 74AS and 74S series). The machine could be drastically size reduced by conversion to VLSI, chip sets, but unfortunately these usually employ some variant of M.O.S. technology which tends to be lower speed than the Schotkey and advanced Schotkey TTL. However by the same argument that supported dynamic ram random access frame stores, a speed degradation may be acceptable for some applications and considering the economics of manufacture is likely to be forced upon more speed conscious tasks as well. Custom integration is therefore a feasible step if volume production is anticipated.

At the heart of the VOS is the SMDU. This is rather over elaborate for a final system which although still requiring the state machine sequencer would not need its development support. This would mean moving away from the ram based synchronous state machine towards registered Pals or Proms

Currently the VOS addresses memory in patterns dictated by the contents of a RAM databank (not to be confused with SMDU RAM), which for variable boundary and special purpose dumps is rather a chore to fill. To overcome this a number of switchable look up tables could be provided for the more commonly used irregular window shapes which would make them as easy to use as the fixed boundary ones. In some circumstances this would reduce the dump set up overhead and may go some way towards compensating for derated transfer speeds.

Currently when dumping to a window only the lower 12 bits are transferred and so the upper 4 bits are random garbage. By a modest modification it would be possible to hold these locations at their values prior to the dump, or optionally clear them. Therefore if some process on a window yield a binary image this could be marked in the upper bits prior to the next dump. In that way a window could hold the dump data plus 4 binary images (or some other flags). This would be a useful capability for multiple pass functions).

As far as major modifications to the key operations of the VOS transfer system are concerned, they should not be attempted by anyone with less than total knowledge of the system. Nevertheless new modes of operation are permitted providing they use, rather than modify, existing timing cycles. One tentative option that springs to mind is to provide slave to slave dumps without routing

through the frame stores.It must be emphasised that currently there is no crucial requirement for this, but it is mentioned because observation of the current system shows that at present it is almost a possibility. In broadcast dump mode it is possible for slaves on both S&T busses to receive the same dump data, thus providing a common link between the slave databusses. If then the store output is forced tristate and one slave to read the other to write a transfer between window is feasible, providing the VOS can be modified to produce suitably phased addresses. If this were practical this suggests the radical step of discarding separate frame stores and replacing them by dynamically positioned windows, although this is likely to be too adventurous to include in a 1st production system.

Whilst on the subject of modified transfers/accesses it might be desirable to provide a direct access mode to the frame stores. This might be motivated by a debug requirement, where memory could be tested without recourse to the VOS machine. Alternatively it might enable the frame stores to be sold without the rest of the system. Fortunately control lines already exist to control store access, but either the VOS or Video scanner would need modifications to provide direct store addresses. Furthermore if multiple processor access is required then contention handling will be necessary.

10.5. "TURBO CHARGING" THE DTP

This unit has proved to be an invaluable asset to VOSTTAC1, providing movement detection, averaging, edge detection and thresholding capabilities all at very high speed. At present each store card houses a DTP unit, as this was the most convenient way to incorporate the features without disturbing existing capability.

From this it should be gathered that VOSTTAC1 has already undergone one retrofitting phase prompted by a desire to accelerate identified key processes. Because of its delayed arrival on the scene there was only physically room for a few circuit blocks on each store and an interstore communications link was necessary, somewhat limiting its potential capability. If an uprated system is considered there are considerable advantages in placing a new enhanced DTP between the store cards and slaves. This would not only do away with the interstore communications link, but also by relaxing the space constraints, would allow a much more powerful DTP to be implemented. Bearing in mind the enormous advantages of the minimal DTP a larger in line version could have phenominal benefits.

10.6 SYSTEM SOFTWARE MODS

The system software enabled the machine to perform to its full capabilities - realised as a suite of machine code routines under the control of a simple executive. To the experienced system programmer this provides a framework for developing high speed application programs. For less competent programmers a higher level of abstraction would be desirable, but also introduce speed reductions. It can be argued from a technical viewpoint that the performance loss is intolerable but on the other hand to make the system appeal to as many customers as possible a higher level language must be considered. The existing system uses TMS32010S which although powerful signal processors are somewhat lacking in general instructions which make it difficult to realise desirable loop and branch constructs without wasteful storage and retrieval of variables. Coupled with the small program memory and the inclusion of internal datamemory, which must be unusual to compiler writers, a high level language on TMS32010'S is unlikely. Consequently the calling of established routines is likely to be the highest level of user control for VOSTTAV1. The TMS32020 is a different matter entirely with its larger addressing range and more complete instruction set. If the device was used it would be unwise to use a high level non specific application language, i.e. BASIC or FORTRAN. Much better to use something lower level like "C". It is much easier to imagine how a compiler could fit around the machine code instructions and so is likely to produce reasonably efficient code.

A language which is attracting a great deal of interest is the ANSI standard 'ADA' (ref. 99-100). This is a multitasking portable modular language which is the DOD standard (USA Defence) and will be the MOD standard after 1st July 1987. Early compilers were self hosted although cross compilers to target machines are becoming available. It should be borne in mind that ADA software tools are expensive, which must be partly due to the rigorous standards tests before ADA type approval. In addition, there is not as yet widespread expertise at using the language in the UK. Even if compiler support did appear for the TI chip it would need to do so in conjunction with an embedded operating system for the target system, e.g. VRTX (ref. 101). Because of the difficulties mentioned it is unlikely that language support will be provided for the less main stream processors such as the TMS32020 at least in the near future.

Program storage within a VOSTTAC production system would be prom based, as the software development phase should have passed. This would leave the slaves with a library of tasks suitable for the various applications considered. The master prom would contain the necessary executive routines to control the machine to perform in its specified role.

As applications become more ambitious and complicated to control it is likely that the limiting addressing range of the master will prove inadequate. To alleviate this problem paged "prom banks"

could be used, however this is not a natural extension and requires a fair amount of "juggling" to produce the software using existing tools. This tends to support the case for considering a TMS32020 as the eventual system master.

VOSTTAC1 encorporates a degree of system error checking both at the hardware and software level, either signalling fault conditions or rerouting a particular process. Considering the valuable benefits from this minimal monitoring system the inclusion of a more powerful and comprehensive diagnostics facility would seem very desirable indeed. Not only would the whole machine be more robust in operation but the debugging and commissioning of circuit modules would be greatly simplified.

10.7. MISCELLANEOUS IMPROVEMENTS

A general purpose piece of image processing kit when offered for sale must be fairly robust and not fall apart the first time somebody nudges its casing. Therefore the method of construction must be mechanically sound. The prototype system was wire wrapped on boards with integral earth planes, which provide an untidy looking mass of wires. Asthetics apart, this connection method can have quite good electrical characteristics if random wiring is adopted to reduce crosstalk. On the other hand the mechanical problems can be dire, as physical impact need not be large to cause wire wrap pins to short together. In addition intensely

wired boards have a tendency to distort under the combined "pulls" of all the wires, and when inserting in a rack the opposing forces which straighten the board can stress and eventually break the connections.

The obvious solution is to transfer the whole system to printed circuit boards. This step is only advisable when the final design is firmly defined, as the complexity of the boards will demand multilayer construction which is too expensive to experiment with.

The history behind the auxilliary display output of VOSTTAC1 was that it was offered out as a separate short term development project (ref. 28) and consequently suffered from limited objectives. It was really only suitable for displaying moving markers, error messages, and of course performing the target nomination function. As the possible system applications became more diverse it began to be apparent that it would be beneficial to enhance the auxilliary output in terms of better graphics and logging facilities. This might take the form of a cheap microcomputer with perhaps a disc drive and sufficient software to perform a variety of requested tasks. A reasonable requirement for the main system would be the provision of a general purpose interface so that several different types of computer could be used as display peripherals. Furthermore such a connection would allow a means of transfering <u>colour</u> images into computer stores for off-line analysis , a facility not normally offered i.e.

luminance only inputs are the norm. In passing it should be mentioned that picture storage retrieval on the BBC model B is already possible using a simple existing interface.

One last point is that as image processing systems might be needed in all sorts of awkward positions and environments, the option of battery powering should be supported.

The foregoing deliberations have put forward ways in which VOSTTAC1 might be physically evolved towards a practical production system. Some of these involve a high risk, i.e. major changes whereas others are not necessarily essential. The changes which must be considered are summarised below:

a) PCB Construction

b) Dynamic Ram Option

c) Flexible Video Front End

d) In Line DTP

These changes are medium risk as they involve variations on currently existing capability. (It should be noted that c&d might offset the extra overheads due to b).

CONCLUSIONS

As a tool for investigating high rate image analysis VOSTTAC1 has surpassed expectation. With it, problems may be tackled, within the Electrical Engineering Department of Bath University which were hitherto impracticable. The system has been recognised at international conference level not only for its image analysis capabilities (ref. 22) but also as a microprocessor application in its own right (ref. 21). The Vector Orthogonal Scanner is currently the subject of a patent investigation which will hopefully lead to a full patent application. In addition images produced by VOSTTAC1 have been displayed at a Royal Photographic Society exhibition in Bath.

Considering digital video systems the machine incorporates frame stores whose random access speed is unlikely to be rivalled by any conventional image store and can be filled with data derived from any PAL or NTSC source.

To process image data there is the 25 Million Instructions Per Second (MIPS) capability of the parallel multiprocessor consisting of (up to 5 in the MKI version) special purpose 16 Bit signal processing devices with powerful instruction sets.

The digital video and concurrent processing systems are embedded within a painstakingly designed architectural environment which above all preserves the flow of visual information throughout the machine.

Data transfers within the system are handled by the Vector Orthogonal Scanner (VOS) which has the power to reduce 2-D spatial problems to tests on 1-D sequential datastreams.

This can not only decrease addressing overheads but drastically reduce processing times. The pipelined hardware has the rare, if not unique ability to scan frame stores along angled vectors at very high speed. Furthermore it is possible in some circumstances to simultaneously transfer data derived from two orthogonal scan directions. These properties can be used to build angular independence into algorithms at a very early stage. The vector scanning shape descriptor is an extension of this and yields an attribute which is size and orientation independent, has a noise averaging property, yet yields both area and skew as secondary features. Moreover the shape coding is data compressional thereby providing an extremely quick algorithm which is more flexible and requires less storage than conventional template model comparisons.

The system has been shown capable of performing many low level image processing techniques, which where applicable have been accelerated by firmware in the form of the Data Transfer Processor (D.T.P.).

A DTP uses very little hardware and programmable logic, has no memory as such although can produce a delay, is ineffective unless data is flowing through it and yet is a phenominally powerful unit. Using DTPs it is possible to produce throughline moving picture displays which would otherwise be impossible and in any case rarely seen on other image processing systems.

Colour image analysis is uncommon yet supported by VOSTTAC1, which has demonstrated the region segmentation power of multispectral thresholding. Special purpose hardware is provided (DTP) which allows through line, window adaptive, thresholded displays to be viewed and yet encorporates level mapping which greatly eases software pixel tests.

Various edge enhancement techniques have been demonstrated, practical applications investigated and highlighted lines coded by the method proposed by Hough. Furthermore it is possible to display in real time edge enhanced images using a convolution circuit in the DTP. This allows camera scenes and video taped recordings to be viewed in edge as opposed to raw image space. What is even more remarkable is that because vector scanning is permitted, directional differentiators need not be applied along

horizontal and vertical directions. For example it is quite permissible to perform the previously unheard of tasks of using say a Prewitt mask directly along a diagonal. The filtering is not restricted to edge enhancement and can be extended to include averaging and low pass filter functions.

Background subtraction and hence movement detection is available as another DTP primitive which allows such images to be displayed at frame rate divided by two, due to the necessity of two input images for every output picture. This is an invaluable facility for surveillance and inspection systems and again is not commonly available at the rates achieved. In common with other DTP functions none of the software signal processing time is required and so is free for other tasks.

VOSTTAC1 can track visually observed targets at frame rate and can outperform a human at the same task. It must be exceptional to find a machine capable of fairly general image processing use, within a University environment, that can track at such a high rate. Furthermore the processing has been shown extendable to multiple targets and has by no means reached the upper bound on its capabilities.

A vast range of potential application areas has been revealed by the MKI system allowing future investigations to be pursued along a broad front. With anticipated commercial interest the possible evolution of a VOSTTAC based production unit is an exciting prospect.

Although the initial research period, journaled in this thesis has now concluded a follow on study is already in motion. Current areas of interest involve the further investigation of multi target tracking, the commissioning of an articulated camera platform under system software control and the design of a general purpose interface for the IBM PC.

Finally then, the pioneer work on the system can perhaps best be described in a melodramatic fashion.....

"VOSTTAC1, it came, it saw and its still conquering!...."

APPENDIX (A)

SENSORS & RANGING SYSTEMS

APPENDIX (A)

SENSORS AND RANGING SYSTEMS

Any wavelength of the electromagnetic spectrum could in theory be used in a particular sensing device, however in practise severe attenuation during transmission through the atmosphere limits or prevents the use of some regions.

Conventional radars operate at the microwavelengths from greater than 10cm wavelength down to 1cm. Beyond this lie the millimetre and submillimetre regions for which the atmosphere is highly non transmissive until a window is reached in the FIR region (8-14 um). Another atmospheric window occurs in the MIR region (0.7 - 2.5 um). Infra red detectors tend to be expensive, using electromechanical scanning and requiring cryogenic cooling. Some pyroelectric detectors do exist which operate at room temperature but their performance is inferior to the cooled devices. Transducers operating in the NIR or visual regions, such as conventional and image intensified TV cameras are simple and inexpensive but of course suffer from adverse atmospheric conditions when used outdoors. Although this class of passive sensor is of primary interest to this research effort it is of interest to consider active transducers in slightly more detail. Active systems transmit radiation into their surroundings and measure any reflections. Examples for surface use include

conventional radar (used extensively for medium and long range tracking), millimetre and submillimetre radar and more recently scanning laser rangefinders. The interest in the higher frequency radars can be appreciated by considering the expressions for R.M.S errors associated with radars operating at the microwave bands.

$$\text{Angular error} \quad \sigma e = K \times \frac{\theta}{\sqrt{2 \times SNR}} \qquad (98)$$

$$\text{Range error} \quad \sigma\tau = C \times \frac{T}{2} \times (1/2\ SNR)^{1/2} \qquad (99)$$

where

C	=	speed of light
T	=	pulse width
SNR	=	signal to noise ratio
K	=	constant
θ	=	diffraction limited beam width

By using shorter wavelength radiation, narrower pulse widths (approx 10nS) can be genrated which reduces both range and angular errors.

Scanning laser range finders are attracting interest as they can produce a depth map of a scene which is an ideal "launching pad" for 3-D analysis and physical object detection/avoidance. A major

problem of this type of sensor which is not easily resolvable is the ability to scan out a solid angle governed by the relationship.

$$T_S \quad \alpha \frac{\Omega}{\lambda} \qquad (100)$$

where

λ = wavelength

T_S = scanning time

Ω = solid angle

Therefore the smaller wavelength which reduces range and angular errors, increases the time taken to scan out a solid angle. This effectively limits the magnitude of the solid angle that can be scanned out in a given time.

At present such ranging systems are available in the NIR, MIR and FIR for which the predicted effects of atmospheric conditions are summarised in FIG (61). Millimetre and submillimetre radar is persisted with despite atmospheric attentuation mainly because of its good all weather characteristics. However current work in this area is inhibited due to the shortage of components capable of functioning in the GHZ range.

APPENDIX (B)

THE 2-D FOURIER TRANSFORM

APPENDIX (B)

THE 2-D FOURIER TRANSFORM

The Fourier transform is an invaluable signal processing tool. Unfortunately its application to 2-dimensional images of even modest proportions results in lengthy computation. If off-line image analysis is considered and time penalties are acceptable, if inconvenient, then the calculation of the Fourier Transform might well be a desirable step. Whilst there should be little trouble in finding practical information about the ubiquitous 1-dimensional transform its extension to 2-dimensions is lacking from many texts. To overcome this deficiency the basic equations are noted below along with a suggested practical method for obtaining the 2 dimensional transform from 1-dimensional transforms.

The 1-D Linear Fourier Transform

$$\mathcal{F}\{f(x)\} = F(u) = \int_{-\infty}^{\infty} f(x) \exp[-j2\pi ux]dx \qquad (101)$$

The 1-D Linear Inverse Fourier Transform

$$\mathcal{F}^{-1}\{F(u)\} = f(x) = \int_{-\infty}^{\infty} F(u) \exp[j2\pi ux]du \qquad (102)$$

The 1-D Discrete Fourier Transform

$$F(u) = \frac{1}{N}\sum_{x=o}^{N-1} f(x)\exp[-j2\pi ux/N] \tag{103}$$

The 1-D Discrete Inverse Fourier Transform

$$f(x) = \sum_{u=o}^{N-1} F(u)\exp[j2\pi ux/N] \tag{104}$$

The 2-D Linear Fourier Transform

$$F(u,v) = \int_{-\infty}^{\infty}\int f(x,y)\exp[-j2\pi(ux+vy)]dxdy \tag{105}$$

The 2-D Linear Inverse Fourier Transform

$$f(x,y) = \int_{-\infty}^{\infty}\int F(u,v)\exp[j2\pi(ux+vy)]dudv \tag{106}$$

The 2-D Discrete Fourier Transform for a NXN Array

$$F(u,v) = \frac{1}{N}\sum_{x=o}^{N-1}\sum_{y=o}^{N-1} f(x,y)\exp[-j2\pi(ux+vy)/N] \tag{107}$$

The 2-D Discrete Inverse Fourier Transform for a NxN array

$$f(x,y) = \frac{1}{N}\sum_{u=o}^{N-1}\sum_{v=o}^{N-1}F(u,v)\exp[j2\pi(ux+vy)/N] \tag{108}$$

An important property of the 2-D transform is that it is separable, i.e. it can be calculated by repeated application of the 1-D transform. The equations may be written thus

$$F(u,v) = \frac{1}{N}\sum_{x=o}^{N-1}\exp[-j2\pi ux/N]\sum_{y=o}^{N-1}f(x,y)\exp[-j2\pi vy/N] \tag{109}$$

$$f(x,y) = \frac{1}{N}\sum_{u=o}^{N-1}\exp[j2\pi vx/N]\sum_{y=o}^{N-1}F(u,v)\exp[j2\pi vy/N] \tag{110}$$

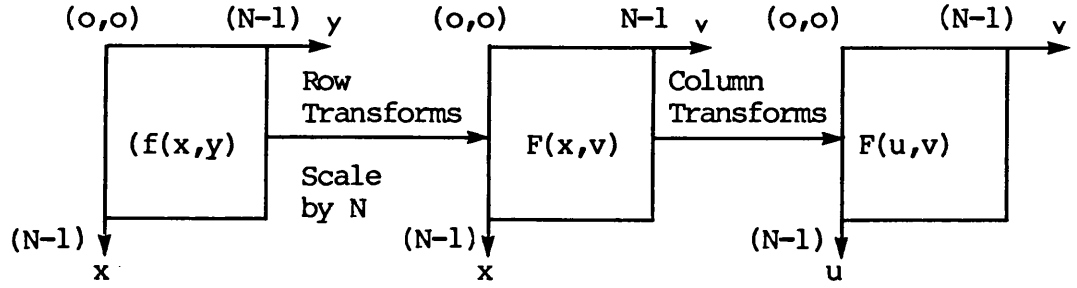
Equation 109 may be written another way to make the separability more obvious, i.e.

$$F(u,v) = \frac{1}{N}\sum_{x=o}^{N-1}F(x,v)\exp[-j2\pi ux/N] \tag{111}$$

where

$$F(x,v) = N\left[\frac{1}{N}\sum_{y=o}^{N-1} f(x,y)\exp[-j2\pi vy/N]\right] \qquad (112)$$

Therefore it can be seen that the 2-D function F(x,v) is obtained by transforming each row of f(x,y) and scaling the result by N. The 2-D Fourier transform is then produced by transforming along each column of F(x,v) e.g.



A similar procedure holds for the inverse transform.

The properties of the 2-D Fourier transform are presented in lucid fashion in ref. 85 along with a simple routine for obtaining the 1-D Fourier transform.

APPENDIX (C)

PUBLISHED PAPER 1

MAYES K.E., MACCORMAC J.K., 1985
"THE DESIGN OF A VECTOR SCANNING, PATTERN RECOGNITION SYSTEM, WITHIN A TMS320 MULTIPROCESSOR ARCHITECTURE".
FOURTH SYMPOSIUM ON MICROCOMPUTER & MICROPROCESSOR APPLICATIONS, BUDAPEST HUNGARY, p. 17-26.

The Desing of a Vector Scanning, Pattern Recognition System. within a TMS320 Multiprocessor Architecture.

Mayes K.E. Maccormac K.
University of Bath
England

This paper illustrates some important considerations during the design phase of an on-line pattern recognition system.In particular the characteristics of the individual processing element (P.E.) are examined in order to create an environment wherein possible data path constrictions are avoided and the more powerful processes optimized. Current research has led to the design of a Vector-Orthogonal-Scan unit and elementary algorithms are investigated.

1.INTRODUCTION

Image processing within rigid time constraints,creates heavy processing demands,so it is fortunate that such analysis is readily divided into sub tasks suitable for parallel techniques.It would therefore seem well justified to adopt a multiprocessor system as opposed to a single centralized unit.However the choice of P.E. is by no means as obvious and seemingly subtle differences between devices may assume immense importance when embedded in a final system.In fact there are very few formalized criteria for P.E. suitability,the most notable being the provision of the binary logical operators[1] and fast cycle times.Satisfying these constraints may be considered as a tuning procedure from which peak processes may be identified and associated hardware built.This was the case with the development of the Vector-Orthogonal-Scan (V.O.S) unit,designed to minimize software overheads by producing image rotational and data reduction effects during memory transfers. The transformed data is then in a sequence suitable for signal processing,which is directly applicable to the strengths of the TMS320 microprocessor.

On the basis of the foregoing considerations the TMS320 was thought suitable for evaluation within a complete pattern recognition system,as described in the following sections.

2. OVERVIEW OF THE TMS320

The Texas Instruments' TMS320 16/32 bit microprocessor[2] provides a practical alternative to bit slice design for high speed data processing.It can achieve a very respectable 5 M.I.P.s and exhibits features in common with both array processors and control devices.The high speed can be attributed to efficient pipelining methods within a modified Harvard architecture.This architecture is characterized by having separate program and data areas,allowing a complete overlap of the instruction fetch and execution.A slight variant on the Harvard approach which enhances the system flexibility,is the provision of software instructions which effect word transfers between program and data memories.This useful facility is just part of a powerful instruction set.Single instructions are equivalent to several operations on general purpose machines and are ideally suited to sequential signal processing problems.However it must be emphasized that more conventional instructions are provided including the binary logical operators,which support the broad range of data tests and manipulations necessary for image analysis.Certainly from a programming point of view the TMS320 seems acceptable,but in practice the minimal P.E. would be inadequate when applied to pattern recognition[3].This is because any such system will be very sensitive to data I/O problems as image analysis deals with large amounts of information and the TMS320 cannot directly address external memory.Fortunately the problem may be overcome by means of a modest expansion detailed in the next section and henceforth reference to a P.E. will include the expansion.

3.PROCESSOR DATA I/O EXPANSION

Although the TMS320 cannot produce direct addresses for external data memory, an expansion consisting of auto increment and decrement counters,can be implemented on the I/O ports provided.The P.E. is then optimized for sequential access to a RAM "window" area,compatible with the more powerful signal processing instructions of the TMS320.Sequential tests are required for various techniques,for example,Fourier transforms and correlation[4],however many pattern recognition algorithms,such as clustering and neighbourhood tests[5],require random access to window memory.Therefore this mode is supported,but incurs a small time penalty relative to the sequential option,although transfer times are still comparable with modern microprocessors running at much higher clock rates.The expansion circuit remains efficient providing the window is fairly small (i.e.8 to 16K) which is obviously inadequate for direct access to the large amounts of memory associated with picture storage. It is therefore vital that an efficient memory transfer and management unit be used.

4.MEMORY MANIPULATION AND MANAGEMENT

It has been mentioned previously that the windows are considerably smaller than the frame stores and that their addressing is optimized for sequential access. This necessitates the use of a hardware unit which can effect data reduction whilst maintaining the sequential ordering.In order to achieve these goals,image storage within the frame stores must be formalized.For research purposes a standard PAL video source is used which produces one complete frame from two interlaced fields. Now to ensure continuity of data,like lines must be adjacent in memory space i.e. the fields must be effectively interlaced in frame store memory after digitization. In this way the transfer from frame store to window is simplified and by providing a variable resolution feature the window data may assume greater geometrical importance if required.Dumping from a predefined image area using a horizontal line scanning technique will optimize the window data for tests which search for horizontal trends.Similarly by scanning parallel to a vector,allows trends along that vector to be analysed[6].The geometric significance of this is that the image is apparently rotated without introducing software overheads and whilst maintaining a sequential data stream.Producing the dumps at high speed not only solves system memory management problems,but provides a transform for use in vector scanning algorithms.The physical interpretation of this is the V.O.S. unit,discussed in the following sections.

5.VECTOR-ORTHOGONAL-SCAN

The conventional PAL television system uses a horizontal line scanning technique to encode picture elements and consequently data appears sequentially along a scan line.This PAL standard only applies up to the point when the image is stored within a frame store and providing the store supports random accesses,image dumps to windows may be performed by scanning parallel to a nominated vector.This technique may be visualized as a hardware geometric transform allowing trends to be analysed along vector directions using efficient sequential data algorithms,the features of which are best illustrated by a simple example.

Consider the image of FIG(1a) and for the time being ignore effects relating to finite picture resolution.Scans parallel to various vectors would produce window fills similar to those shown in FIG(1b-1e).Tests could then be applied to each window (such as counting the number of active pixels on a line,say) and a vector of results formed for each of the window orientations i,e.

$$VEC1, VEC2, VEC3, \ldots . VECN$$

Where N is the number of different scan orientations.This may be considered as a learning procedure to generate an attribute matrix M_A.

$$M_A=[VEC1\ VEC2\ VEC3 \quad VECN]$$

Now if the object is moved and rotated,reapplication of a learning test along a single vector will yield a new result VECX.If VECX is correlated with VEC1 to VECN in turn and a match is found against an error criteria to one or more vectors then there is a possibility that the object has been recognised.However as the vectors are geometrically related the results of additional vector scans can be predicted and better confidence levels achieved.It should be noted that a high overall correlation would not only identify an object,but also estimate the skew[7] relative to the normal position.

The foregoing analysis is degraded in practice due to problems associated with the finite bit mapped display.Scanning along vectors other than the horizontal results in a change in geometric distance between the centres of active pixels.This may be viewed as a change of sample rate with orientation,the effects of which may be reduced by applying fixed weighting coefficients to the derived result vectors. Despite the practical limitations the vector scanning and correlation tests are well suited to the TMS320 and therefore execute efficiently.However it must be appreciated that practical window tests may search for multiple attributes and necessitate more correlation tests.The increased processing demand may be spread amongst parallel processors.The limiting factor of window dump speed is inherently fast by design,allowing multiple vector scans well within the system time constraints.

As yet the specification of an area of interest within an image has been overlooked,although a machine has been developed which allows the boundary of an object to be specified.The boundary may be the perimeter of a complex shape or a standardized enclosure such as a square or circle,which may be nominated either by man-machine interface or as a result of previous processing.If the area of interest is restricted to a rectangle it is possible to fill two windows simultaneously, one with the vector scan,the other with a scan orthogonal to it.This V.O.S. technique performs two geometric transforms on image data during the one window fill time,shown in FIG(2a-b).The rectangular constraint is necessary to obtain simple formatting of window space,which effectively provides destination addresses for all the image data.The orthogonal scanner then manipulates these destination addresses to fill a secondary window with data apparently derived from a scan 90° to the actual scanning vector.

The V.O.S. facilities are intended as powerful utilities which may complement or be enhanced by existing algorithms when embedded in a complete system.

6.SYSTEM FRAMEWORK

The pattern recognition system used in this research is represented by the block diagram of FIG(3).Basically the system digitizes and redisplays video information,whilst allowing digital processing on image data.These functions are required to implement a control loop around an external unit via optical means.

Data storage is provided within the two multiplane frame stores provided,one being analysed whilst the other is updated with data from flash converter ADCs, which is then redisplayed on a video monitor via video DACs.Each of the stores can accommodate 9 to 12 planes of memory allowing 3 to 4 bits per colour (RGB).The 512 to 4096 colour combination capability is the result of subjective optical considerations ,weighing image improvement against depth of store ,in contrast to the choice of individual plane size which is dictated by the bandwidth of the optical transducer.In fact the adopted resolution of 224,000 (approx.) pixels is directly related to a Hitachi solid state colour imaging device,chosen for it's suitability for practical applications.At present the RAMs used within the frame stores require very fast access times due to a random access constraint introduced by the V.O.S. machine when dumping to windows.The exact number of processor window pairs required for analysis is likely to be variable and will depend on not only algorithm complexity,but the hardware interactions within the multiprocessor computer.Therefore the architecture of the parallel processing sub system must be formalized.

7.MULTIPROCESSOR ARCHITECTURE

It has been suggested in the past that pattern recognition systems can be treated conceptually as cones or pyramid structures[8] as shown in FIG(4).The most notable feature of which is the decrease in resolution associated with an ascent of the pyramid.The structure supports three general modes of operation,

a)Data reduction (ascent of cone)
b)Iteration (fixed level of cone)
c)Projection (descent of cone as a result of a, higher level strategy decision)

Within this framework modular parallel processing algorithms have been developed successfully by other research groups,which would seem to recommend the general pyramid concept.However direct hardware implementation would restrict P.E.s to fixed layers and hence a reduced number of P.E.s would degrade resolution as only the upper layers of the pyramid could be filled.In practice it may be more desirable to trade speed rather than resolution against hardware,as the former not only allows lower initial cost and effort,but also strengthens the system as fault orientated rerouting could be supported.Therefore a flexible hardware adaptive architecture is required.The chosen system may be considered as an expandable

tree structure,as shown in FIG(5).At the top of the tree is the master processor, responsible for the ultimate control of the slave P.E.s and peripherals on the lower branches,although there is also likely to be a hierarchy of sub masters within the system .Data can reach a particular node by one of two routes,either from the immediate sub master or from the frame stores via the V.O.S. unit.The latter is not penalized by tree height as it can supply data(with optional resolution change) to all levels of the tree simultaneously.

Communications protocol is intentionally simple.Each slave P.E. requires a control vector from a master to initiate one of a library of sub tasks,after which operation is autonomous until end of task is signaled and result transfer is via dual ported RAM.Because of this mode of operation it is possible to use a combination of different P.E. types and hardware within the general framework.The inherent advantage is that new designs can be incorporated without redesigning the whole system,thereby providing a suitable basis for on going research and development.

8.FURTHER ANALYSIS AND APPLICATION

Image identification is possible within strictly controlled environments,using comparitively simple algorithms however the more general scenarios are full of natural distractions.This background "clutter" may be considered as a noise effect to which algorithms must be immune.As a result software tests are generally developed via heuristic approach and are application dependent.

The goal of this particular research effort is to effect control of a physical rig as a result of identification and tracking of an object moving relative to a mounted camera ,under varying degrees of background clutter and image degradation. As a parallel activity the hardware will be periodically revued as regards improvements to suit a wider range of practical applications.

9.FUTURE DEVELOPMENTS

The existing hardware provides a framework within which experimental techniques and algorithms can be evaluated.The physical realisation of the system is not ideally suited to practical control applications and certainly from a production point of view,streamlining and minaturization must be considered.

Memory reduction is an area where great savings in terms of space,cost and power consumption can be made.At present fast static RAMs are used,due to a random access speed constraint.If it were found that this requirement could be relaxed or alternatively,if there were a significant advance in technology,then dynamic RAM modules could be used for system memory.Another more drastic reduction strategy would discard the conventional frame stores and replace them with dynamically allocated windows.The V.O.S. unit then performing window to window transfers.

The PAL encoder/decoder pair is an obvious inefficiency and should eventually be replaced by direct scanning of the optical imaging device.A desirable refinement would allow alternative sensors such as infrared to be supported.In addition the scanned data could be enhanced by the use of preprocessors in the form of programmable array logic,the facility for which is also included in the present system.

New advances in technology should be incorporated within the system where possible and logical circuits,especially the V.O.S. unit should be commited to VLSI.

10.CONCLUDING REMARKS

In this paper the design phases of a practical on-line pattern recognition system have been illustrated.Particular emphasis has been placed on producing flexible hardware which is compatible with a range of image analysis algorithms. A prime example being the V.O.S. unit which was designed to overcome data flow inadequacies,yet promises to be a powerful and flexible memory transfer tool in the future.

11.ACKNOWLEDGEMENTS

The author would like to thank the University of Bath and Honeywell Leafield Ltd. for their continuing support of the on-going research project.

12.REFERENCES

[1] M.J.B.Duff,The Elements of Digital Picture Processing,in:Onoe,Preston and Rosenfeld /eds./,Real Time Parallel Computing (Plenum 1981).

[2] Texas Instruments,The TMS32010 User's Guide,1983.

[3] V.Cantoni & S.Levialdi,Matching the Task to an Image Processing Architecture, IEEE Proc.6th Patt.Recog.Conf. (1982) p254.

[4] D.H.Ballard & C.M.Brown,Computer Vision,(Prentice Hall 1982).

[5] R.M.Haralick,Some Neighbourhood Operators,in:Onoe,Preston and Rosenfeld /eds./, Real Time Parallel Computing (Plenum 1981).

[6] G.R.Gindi & A.F.Gmitro,Optical Feature Extraction Via the Random Function, IEEE Proc.7th Patt.Recog.Conf. (1984) p702.

[7] S.A.Friedberg & C.M.Brown,Finding Axis of Skewed Symmetry,IEEE Proc.7th Patt. Recog.Conf. (1984) p322.

[8] R.Miller & Q.F.Stout,The Pyramid Computer for Image Processing,IEEE Proc.7th Patt.Recog.Conf. (1984) p240.

FIG.1 SINGLE VECTOR SCANNING

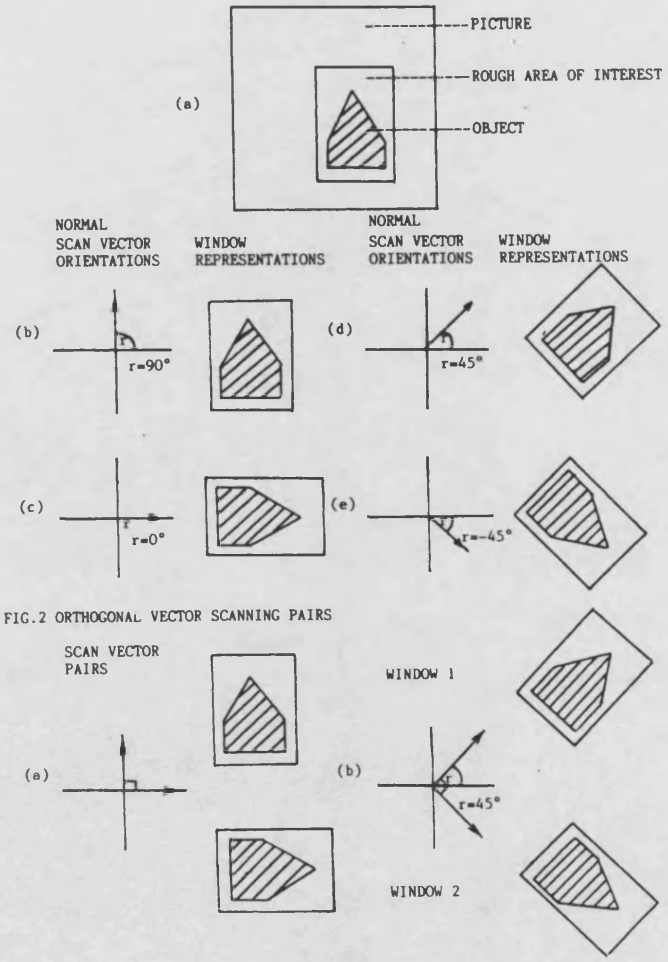


FIG.2 ORTHOGONAL VECTOR SCANNING PAIRS

FIG.3 SYSTEM BLOCK DIAGRAM

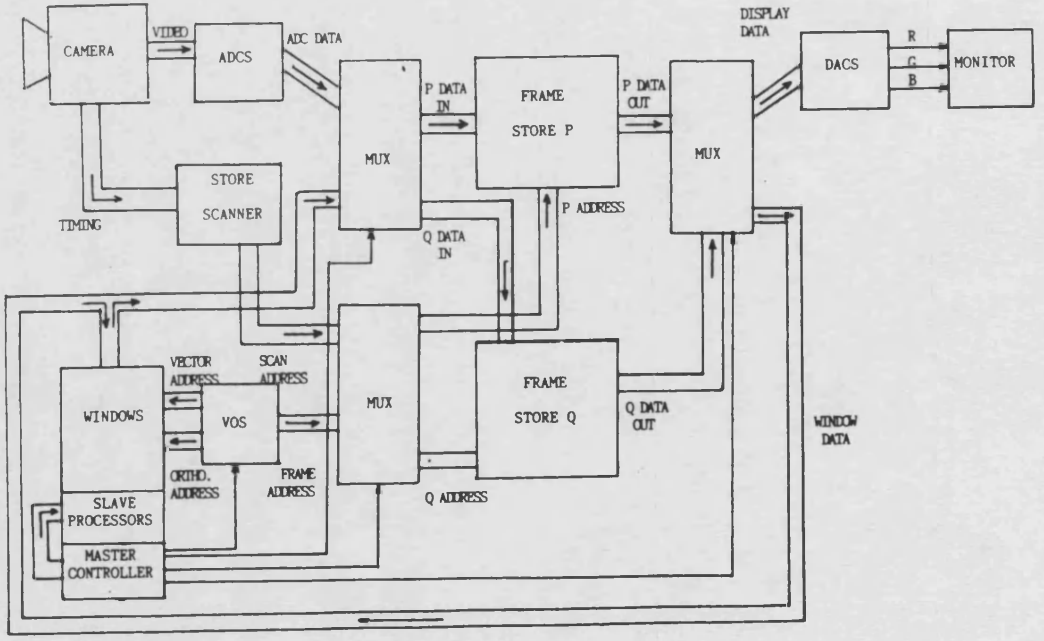


FIG.4 PYRAMID STRUCTURE

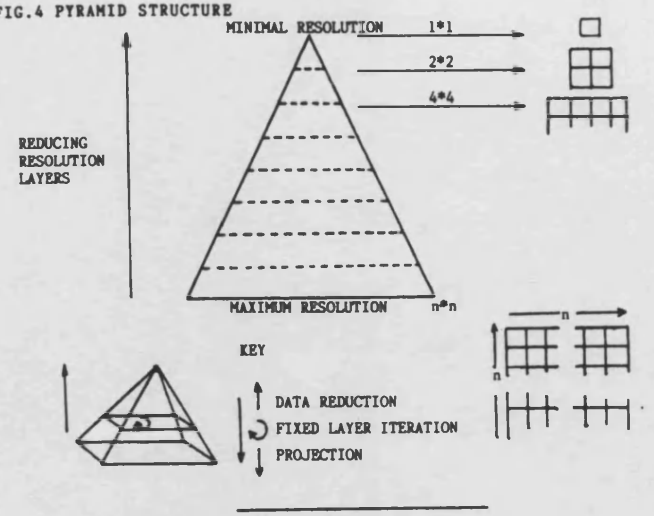
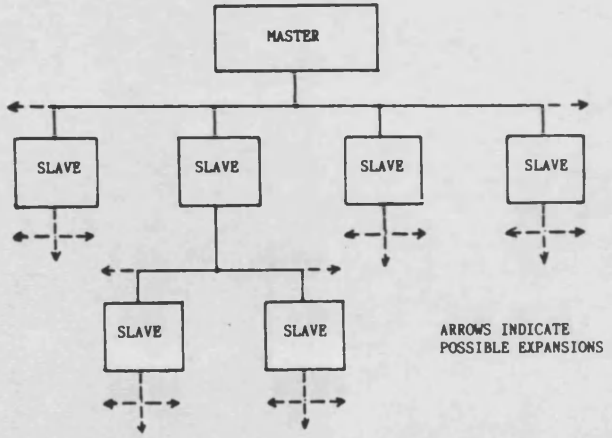


FIG.5 EXPANDING TREE STRUCTURE



HIGH SPEED MICROPROGRAMMABLE DIGITAL SIGNAL PROCESSOR BASED ON BIT SLICED ELEMENTS

K.Tóth, B.Feher, Z.Szalai, Zs.Abonyi

Technical University of Budapest
Depertment of Measurement and Instrument Engineering

This paper describes the SIGNAL-32 general purpose signal processing unit built up with bit-slice microprocessor elements. SIGNAL-32 is a slave processor of the MMT Microprocessor Application System. The hardware architecture of SIGNAL-32 permits the parallel operation of the arithmetic units, which is supported by horizontal microprogramming as well. After introducing the programming and the development environment some application examples are given.

1. INTRODUCTION

Activities in digital signal processing have increased rapidly in the past decade. The application of new procedures requires more sophisticated architecture than the standard microprocessor systems have. The processing of the multichannel

APPENDIX (D)

PUBLISHED PAPER (2)

MAYES, K.E., MACCORMAC J.K., 1986
"VOSTTAC 1, THE VECTOR ORTHOGONAL SCANNING, TARGET TRACKING ADAPTIVE CONTROLLER".
I.E.E. SECOND INTERNATIONAL CONFERENCE ON IMAGE PROCESSING AND ITS APPLICATIONS, IMPERIAL COLLEGE LONDON. p. 171-174.

VOSTTACI: THE VECTOR ORTHOGONAL TARGET TRACKING ADAPTIVE CONTROLLER

K. E. Mayes & J. K. M. MacCormac

University of Bath, UK

INTRODUCTION

This paper is a review of current research effort at Bath University using VOSTTACI, the Vector Orthogonal Scanning Target Tracking Adaptive Controller. The hardware was created as a research tool to investigate practical techniques for visual image analysis which would satisfy the following constraints. The unit would be able to produce real time control signals for target tracking whilst being compact and self contained, suitable for mounting on a small autonomous vehicle. In addition, there would also be the facility for a manual operator interface whereby a target could be nominated by the user.

Examination of the specifications revealed a requirement for a high speed signal processing unit, which supported the continual throughput of video information and allowed a user nominated target learning mode. To perform adequate guidance, control signals should approach frame update rate if possible, i.e. 40mS for a PAL T.V. system. This tracking time constraint necessitates the use of parallelism, however, a massively parallel system would trade off physical compactness against speed and tend to limit the flexibility of approach necessary during the research and development phase. As a consequence, tracking algorithms at least must be limited to matching of easily extractable features, from frame to frame, to allow a more compact computing unit to be used. Filtering is an essential area of image processing and much classical analysis uses Fourier transforms in order to change a convolution into a multiplication. Unfortunately, FFTs are computationally expensive with respect to the tracking time constraints mentioned. If frequency functions cannot be produced and multiplied then the time or spatial functions must be convolved in order to apply any filter transfer functions. The TMS320 reduced instruction set signal processor (1) is very good at one dimensional convolutions and correlations on sequential data. To apply it to two dimensions requires that image data be made sequential along any direction, which can be approximated to by vector scanning of the picture, a technique which, if implemented efficiently, can be used as part of orientation independent algorithms. From the foregoing consideration a hardware system was designed which is detailed below.

VOSTTACI

This machine (2) may be considered as the block diagram of Fig. 1. Basically the video section synchronises to external PAL signals, digitises the incoming colour information and stores/displays it via the 384*577 frame stores to a maximum depth of 12 bits per pixel. The signal processing unit is a tree structured multiprocessor TMS320 unit with a master slave heirarchy. A two level implementation containing 5 processors would peak at 25 MIPs. Each processor is associated with separate window, data and program memories. The window can be filled with data from the framestores using the VOS unit. This machine can download areas within the picture into the window by scanning parallel to a vector. Using an optional line and pixel skipping facility it is possible to produce a reduced resolution effect as may be required for a pyramid (3) type approach. If the area of interest is rectangular there is a bonus, in that the orthogonal scan dump can be broadcast at the same time, i.e. both vertical and horizontal scans produced simultaneously.

The area of interest can be located through software or via the user nomination interface. This takes the form of an auxiliary overlaid screen bit map display with its own processor. The user input is via a light pen, with which an operator can enclose a target of interest within a highlighted boundary and thereby initiate an on line learning phase and subsequent tracking. The auxiliary display is also used as an output utility by the main system.

Control information can pass to and from the real world using the adc, dac, timer boards, each of which has sufficient analogue I/O capability to achieve state feedback control of physical machines with three degrees of movement. The type of tracking control attempted is obviously very dependent on the demand update rate which is subject to hardware and software constraints.

Tracking Limits

An essential criteria for the tracking system is that it should easily satisfy the time constraints when analysing a simple target with a bimodal histogram, using the most elementary algorithm possible. It has been shown that VOSTTACI can track a simple coloured target on a black background using a simple algorithm, whereby 2 processors are used to calculate the centre x, y cords of a local region, existing with a 4K image window. Each processor works on dump data, although the windows are orthogonally related, i.e. horizontal and vertical, which allows the same program to be run on each slave to generate both the x and y central coordinates. Both windows can be filled simultaneously with data because of the orthogonal transfer feature of VOSTTACI. The process is in 2 phases, the first scanning the 64 * 64 window to produce 64 results, the second phase scanning the intermediate results to produce a coordinate value. An approximate measure for the overall processing time can be arrived at as follows:

T = D + M + R where . . . (1)

T = total processing time approximation

D = Dump time < 1mS (for 4K window)

M = Memory scan time = n * n * m * C. . . (2)

R = Result scan time = n * p * C . . . (3)

m = average instructions per pixel = 10

p = average instructions per result = 10

C = instruction cycle time = 250nS

n = width or height of a square window in pixels

For a 64 * 64 window the processing time is approximately 12mS, and for a 90 * 90 about 22mS, using 2 slaves.

From the foregoing analysis and estimates the system can track an object within a frame time under the most

favourable conditions possible. Furthermore a lower limit on any processing is likely to be between 10 and 20mS. If 20mS is assumed, which is convenient as it corresponds to the PAL T.V. field update rate, then the control information rate is at best 50Hz. If it is required to control a machine with frequency 5 rads/S then in practice at least 6 samples are required to reproduce the fundamental, requiring a minimum sample rate of approx. 30 rads/S. The maximum sample period is therefore 200mS or 5 frame times which gives an upper bound to the processing time. In practice, all tracking activity should be kept within a working limit of 2 to 3 frames. This serves to reinforce the requirement that target tracking must be fast and therefore of limited complexity. Fortunately the user nominated on line learning mode of VOSTTACI gives the processing much needed assistance.

Target Nomination

Nominating a target manually is a means of bypassing the initial search and identification by leading directly to a learning phase followed by a track mode. The main advantage of this supervisory mode of operation is that it allows immediate tracking of an object not contained within the limited static knowledge base of the machine. However, it may also be considered as a means of improving the image, signal to noise ratio. One of the main problems for a simple tracker is that of segmenting the image or subimage into target and non target regions. There are many schemes for doing this but the simplest and usually quickest is by thresholding. Obviously a suitable level has to be chosen which is insensitive to illumination changes. One global technique for achieving this is to compute the grey level histogram (4) for the whole image and, assuming it is bimodal, choose the threshold to exist at the minima between the two maxima. Unfortunately the majority of such histograms are unlikely to be bimodal and even when they are they often require smoothing prior to the minima search. In colour systems there are at least 3 spaces associated with R,G,B which allows multidimensional thresholding either globally or over subimages. A Splitting technique can be applied which searches for sizeable peaks within the histograms, finds the histogram with the biggest response, forms thresholds either side of this peak and uses them to segment the region into sub regions. The initial phase of any tracking action would be to segment the image based on a known model of these threshold levels. The advantage of on line learning is that these threhsolds are determined under their working illumination conditions and are therefore likely to be more accurate than a precomputed model. However, if a rectangular area is nominated around an irregular target there will be "noise" present, as the background cannot be controlled in an on line situation. In which case recursive use of the thresholding algorithm may eventually identify the background as a subregion and if in learning mode this subregion might be wrongly interpreted as a chracteristic "patch" on the target. A means of overcoming this is to make the window assume the same outline as the target, in which case the noise effect is reduced. VOSTTACI has the capability to dump from irregular window areas, which may be nominated by a light pen sketch.

It would be naive to expect any tracking action to be infallible and therefore autonomous search and identify algorithms must be provided to recover "escaped" targets.

Shape Coding

Thresholding an image may highlight several subregions and this is likely to be the situation when tracking has been lost and the target needs to be re-identified. Target trajectory history may guide the start of the search but a feature measurement is required to differentiate between subregions. The choice of features must be limited to those which can be identified rapidly with the available hardware. With this in mind desirable features would be independent of range, orientation and illumination.

Area/Range

Finding the area of a thresholded region is fairly simple, in fact the elementary tracking algorithm previously mentioned produces not only the central region coordinates but two area estimates as well. Area is only useful as an attribute if it is qualified by a distance measurement. Even for fixed distance cases an area criteria contains no shape information, although it can be used to reduce the number of subimages of interest. In a practical situation it is unlikely that the range of every region is known and therefore it must be determined from some test. There are many schemes available although the majority assume either binocular imaging systems or auxiliary range sensors, e.g. ultrasonic, laser, infrared, etc. There is, however, a technique suitable for monocular vision but it requires that the camera system be capable of moving physically through a known displacement (5). In the research VOSTTACI is required to control a video camera through 2 degrees of freedom so it is possible to introduce a known angular displacement to the camera, observe the shift between two corresponding image points and calculate the distance using triangulation. The technique is prone to error if the target is itself moving in an unpredictable fashion and assumes that unique points can be identified within time shifted images. The distance measurement, even if frequently updated, is of no use in discriminating between 2 target "snapshots" of similar area (normalised with respect to distance). In which case some boundary descriptions are required.

LOCI OF RADIUS USING VECTOR SCANS

Given a binary valued subregion, it is possible to locate its centre. The distance (r) from this centre to the perimeter at angle θ to the horizontal is the polar coordinate form of a point on the boundary. If radius measurements were taken for an infinite number of angular displacements then the outer boundary would be transformed to a continuous periodic function of radius against angle (see Fig. 2). If the object were rotated normal to the viewer and the operation reapplied the results would be identical except for a shift in angle. In addition providing the measurements are scaled relative to an area or power measure, the results would be similar for different sized objects of the same shape. It would therefore seem from a theoretical point of view that the loci of radius measurement could provide an attribute which is both size and orientation independent. The problem is how to translate this technique to a fast practical algorithm. If we consider a mainly software strategy, there will be much work in calculating the addresses for the various scan vectors. As mentioned previously the TMS320 functions at its best when the data is in sequential form and would therefore incur time penalties using the above approach. For this reason the vector scanner was evaluated for extensive use within the algorithm.

The basic mode of the VOS scanner transfers window blocks between the frame stores and slaves, although this mode is only really suitable for the following scan vectors assuming a 384*577 screen with 4/3 aspect ratio:

0° 90° 45° 26.6° 56.3°

This is because block mode only uses pixels which lie exactly on a particular vector and therefore for certain angles the interpixel distance is large. Weighting coefficients for the angles above can be normalised with respect ot the horizontal interpixel distance to give:

1	0.5	1.41	1.12	1.8

The higher the number the greater the spatial significance of the pixel, which may be considered as a reduction in sample rate. Applied to the loci measurement a weighting factor greater than 2 would significantly degrade the radii measurements. If the radius against angle measurements were taken using only the acceptable angles the function would have nonlinear sampling and few points making it ill-suited as a target signature. Fortunately the VOS machine can operate in line mode which counteracts these effects.

It was decided to sample the radius over 64 points, thereby specifying a measurement every 5.625 degrees. It was found that convenient pixel ratios could be produced to define an average scan vector to within 0.7 of a degree. In the line mode for steep angles the vector is approximated as a piecewise vertical scan and in this way interpixel distance is greatly reduced. However, a displacement error is introduced, which must be kept small as it is not only a measure of departure from the ideal vector but also of the nonlinearity in interpixel distance. It was found that the error could be kept below 1.5 pixels by using either vertical, horizontal or 45 degree line scans depending on the sector. Accepting these error tolerances the VOS machine can produce vector line scans through an object centre, producing sequential dump data. It is then a simple matter to scan this data to find the 2 outer perimeter measurements. Repeated for all angles yields a 64 point function, $F(\theta)$, which is a periodic shape descriptor target attribute, from which:

$$\int_0^T F^2(\theta)d\theta \qquad \ldots (4)$$

can be calculated and used as a scaling factor. The results are then in a form which can exploit the correlation (6) facilities of the TMS320 special instruction set. The autocorrelation function is given by:

$$R(\phi) = F(\theta)*F(-\theta) = \int_{-T}^{T} F(\theta).F.(\theta+\phi)d\phi \qquad \ldots (5)$$

and has a maximum at $\phi = 0$. A property of the function is that:

$$\int_{-T}^{T} R(\phi)d\phi = \left[\int_0^T F(\theta)d\theta\right]^2 \qquad \ldots (6)$$

Therefore if the radii function of an unknown target is correlated with a known radii function, the above relationships can be used as match criteria (see Fig. 3), subject to size normalisation. If the match is successful, then not only is the target identified, but its orientation is known, using the fact that the autocorrelation function is a maximum for zero shift.

If this technique is extended to 3D the processing requirement is proportional to the number of surfaces used to describe the target. In practice, only a small number of target "snapshots" can be known and therefore identification performance is degraded. However, if initial tracking is prompted by the nominator then the most relevant surface can be assumed. Providing the target movement is slow with respect to the tracking rate, the surface model should still be applicable in the next frame, although over a longer period of time the model matching would be likely to fail. This can be counteracted by updating the model by successive reapplication of the on line learning phase. Such a strategy must be used with a performance measurement so that attributes associated with disturbances do not dominate the target model.

Radii Measurement Within Multiregion Subimages

The inclusion of additional small regions within a window has little effect on the area and centre calculation of the major target, however, they could affect the radii measure in much the same way as a noisy boundary. To counteract this a filter function must be applied to eliminate the noise effects. It is then necessary to apply a segmenter to identify all objects greater than a certain size. If in tracking mode detection of more than one object within a window acts as a warning that the algorithm performance is likely to be degraded and is therefore a good point to inhibit relearning. The radii test is still required to identify the valid target; however, this is only accurate if the window only contains a single object. Fortunately, the marking of distinct regions during segmentation effectively transforms the multiregioned window to a number of single regioned windows. These can then be loaded back to little used areas of the screen for vector analysis. This segmentation can be applied to small areas or globally and is suited to multiprocessor operation. Although primarily intended for maintaining single target tracking in the presence of rogue objects the technique may be applicable to relearning multitarget tracking.

CONCLUDING REMARKS

The basic properties of VOSTTACI have been mentioned and a strategy has been suggested for target tracking and identification using the vector scan and correlation facilities available. It must be stressed that the machine is sufficiently flexible to use techniques not requiring vector scans, and with modest modification could even be used in conjunction with alternative sensors. The ultimate goal of the research is to use the machine to control a vehicle. Tracking is of great importance to maintain an area of interest within a field of view largely independent of the vehicle motion. Future work will tend towards extracting structure from real surroundings in order to guide the vehicle along a given road or path.

ACKNOWLEDGMENTS

The authors would like to thank the University of Bath and Honeywell Leafield Limited for their continuing support of this ongoing research project.

REFERENCES

1. "TMS32010 Users' Guide", Texas Instruments, 1983.

2. Mayes, K.E., MacCormac, J.K.M.: "The design of a vector scanning pattern recognition system, within a TMS320 multiprocessor architecture", 1985, Fourth Symposium on Microcomputer & Microprocessor Applications, Vol.I, pp.17-26.

3. Nagin, P.A. et al: "Region Relaxation in a parallel heirarchical architecture", 1978, Real Time/Parallel Computing, Plenum Press, New York & London.

4. Ballard & Brown: "Computer Vision", 1982, Prentice Hall.

5. Itoh, H., et al: "Distance measuring using only simple vision constructed for moving robots", 1984, Proc. Seventh International Conf. on Pattern Recognition, Vol.I, pp.192-195.

6. Castleman: "Digital Image Processing", 1979, Prentice Hall.

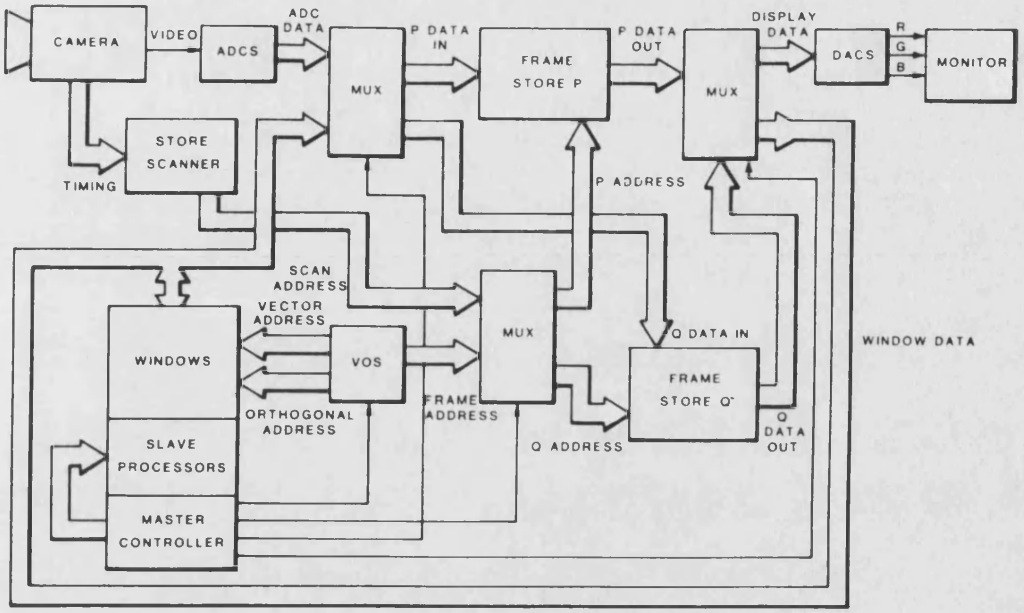


FIGURE 1. VOSSTAC 1 SYSTEMS BLOCK DIAGRAM

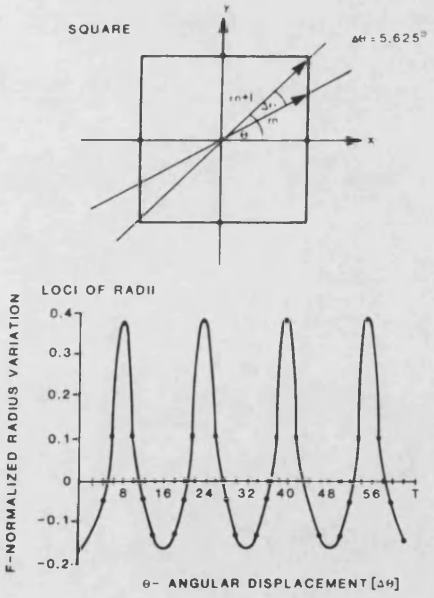


FIGURE 2. RADII FUNCTION OF A SQUARE

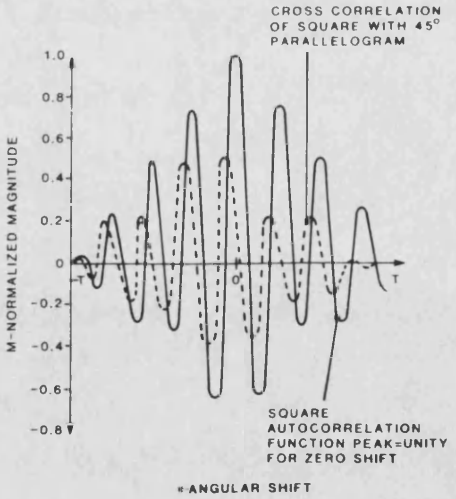


FIGURE 3. CORRELATION FUNCTIONS

APPENDIX (E)

THE TMS32010 INSTRUCTION SET

INSTRUCTION SYMBOLS

SYMBOL	MEANING
ACC	Accumulator
AR0	Auxiliary register zero
AR1	Auxiliary register one (AR0 and AR1 are predefined assembler symbols equal to 0 and 1, respectively.)
ARP	Auxiliary register pointer
D	Data memory address field
DATn	Label assigned to data memory location n
dma	Data memory address
DP	Data page pointer
I	Addressing mode bit
K	Immediate operand field
(N)	Contents of register "N" or data memory location "N"
>nn	Indicates nn is a hexadecimal number. All others are assumed to be decimal values.
P	Product (P) register
PAn	Port address n (PA0 through PA7 are predefined assembler symbols equal to 0 through 7, respectively)
PC	Program counter
pma	Program memory address
PRGn	Label assigned to program memory location n
R	1-bit operand field specifying auxiliary register
S	4-bit left-shift code
T	T register
X	3-bit accumulator left-shift field
→	Is assigned to

Instruction Set Summary

The instruction set summary in the following table consists primarily of single-cycle single-word instructions. Only infrequently used branch and I/O instructions are multicycle.

INSTRUCTION SET SUMMARY

ACCUMULATOR INSTRUCTIONS																			
MNEMONIC	DESCRIPTION	NO. CYCLES	NO. WORDS	OPCODE INSTRUCTION REGISTER															
				15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABS	Absolute value of accumulator	1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	0	0	0
ADD	Add to accumulator with shift	1	1	0	0	0	0	←	S		→	I	←			D			→
ADDH	Add to high-order accumulator bits	1	1	0	1	1	0	0	0	0	0	I	←			D			→
ADDS	Add to accumulator with no sign extension	1	1	0	1	1	0	0	0	0	1	I	←			D			→
AND	AND with accumulator	1	1	0	1	1	1	1	0	0	1	I	←			D			→
LAC	Load accumulator with shift	1	1	0	0	1	0	←	S		→	I	←			D			→
LACK	Load accumulator immediate	1	1	0	1	1	1	1	1	1	0	←				K			→
OR	OR with accumulator	1	1	0	1	1	1	1	0	1	0	I	←			D			→
SACH	Store high-order accumulator bits with shift	1	1	0	1	0	1	1	←	X	→	I	←			D			→
SACL	Store low-order accumulator bits	1	1	0	1	0	1	0	0	0	0	I	←			D			→
SUB	Subtract from accumulator with shift	1	1	0	0	0	1	←	S		→	I	←			D			→
SUBC	Conditional subtract (for divide)	1	1	0	1	1	0	0	1	0	0	I	←			D			→
SUBH	Subtract from high-order accumulator bits	1	1	0	1	1	0	0	0	1	0	I	←			D			→
SUBS	Subtract from accumulator with no sign extension	1	1	0	1	1	0	0	0	1	1	I	←			D			→
XOR	Exclusive OR with accumulator	1	1	0	1	1	1	1	0	0	0	I	←			D			→
ZAC	Zero accumulator	1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	0	0	1
ZALH	Zero accumulator and load high-order bits	1	1	0	1	1	0	0	1	0	1	I	←			D			→
ZALS	Zero accumulator and load low-order bits with no sign extension	1	1	0	1	1	0	0	1	1	0	I	←			D			→

AUXILIARY REGISTER AND DATA PAGE POINTER INSTRUCTIONS																		
MNEMONIC	DESCRIPTION	NO. CYCLES	NO. WORDS	OPCODE INSTRUCTION REGISTER														
				15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LAR	Load auxiliary register	1	1	0	0	1	1	1	0	0	R	I	←			D			→
LARK	Load auxiliary register immediate	1	1	0	1	1	1	0	0	0	R	←				K			→
LARP	Load auxiliary register pointer immediate	1	1	0	1	1	0	1	0	0	0	1	0	0	0	0	0	0	K
LDP	Load data memory page pointer	1	1	0	1	1	0	1	1	1	1	I	←			D			→
LDPK	Load data memory page pointer immediate	1	1	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	K
MAR	Modify auxiliary register and pointer	1	1	0	1	1	0	1	0	0	0	I	←			D			→
SAR	Store auxiliary register	1	1	0	0	1	1	0	0	0	R	I	←			D			→

BRANCH INSTRUCTIONS																			
MNEMONIC	DESCRIPTION	NO. CYCLES	NO. WORDS	OPCODE INSTRUCTION REGISTER															
				15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B	Branch unconditionally	2	2	1	1	1	1	1	0	0	1	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BANZ	Branch on auxiliary register not zero	2	2	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BGEZ	Branch if accumulator ≥ 0	2	2	1	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BGZ	Branch if accumulator > 0	2	2	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BIOZ	Branch on $\overline{BIO}$ = 0	2	2	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BLEZ	Branch if accumulator ≤ 0	2	2	1	1	1	1	1	0	1	1	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BLZ	Branch if accumulator < 0	2	2	1	1	1	1	1	0	1	0	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BNZ	Branch if accumulator ≠ 0	2	2	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BV	Branch on overflow	2	2	1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
BZ	Branch if accumulator = 0	2	2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
CALA	Call subroutine from accumulator	2	1	0	1	1	1	1	1	1	1	1	0	0	0	1	1	0	0
CALL	Call subroutine immediately	2	2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
				0	0	0	0	←			BRANCH ADDRESS								→
RET	Return from sub-routine	2	1	0	1	1	1	1	1	1	1	1	0	0	0	1	1	0	1

T REGISTER, P REGISTER, AND MULTIPLY INSTRUCTIONS

MNEMONIC	DESCRIPTION	NO. CYCLES	NO. WORDS	OPCODE INSTRUCTION REGISTER 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
APAC	Add P register to accumulator	1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1
LT	Load T register	1	1	0	1	1	0	1	0	1	0	I	←			D			→
LTA	LTA combines LT and APAC into one instruction	1	1	0	1	1	0	1	1	0	0	I	←			D			→
LTD	LTD combines LT, APAC, and DMOV into one instruction	1	1	0	1	1	0	1	0	1	1	I	←			D			→
MPY	Multiply with T register; store product in P register	1	1	0	1	1	0	1	1	0	1	I	←			D			→
MPYK	Multiply T register with immediate operand; store product in P register	1	1	1	0	0	←							K					→
PAC	Load accumulator from P register	1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	1	1	0
SPAC	Subtract P register from accumulator	1	1	0	1	1	1	1	1	1	1	1	0	0	1	0	0	0	0

CONTROL INSTRUCTIONS

MNEMONIC	DESCRIPTION	NO. CYCLES	NO. WORDS	OPCODE INSTRUCTION REGISTER 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DINT	Disable interrupt	1	1	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	1
EINT	Enable interrupt	1	1	0	1	1	1	1	1	1	1	1	0	0	0	0	0	1	0
LST	Load status register	1	1	0	1	1	1	1	0	1	1	I	←			D			→
NOP	No operation	1	1	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
POP	Pop stack to accumulator	2	1	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	1
PUSH	Push stack from accumulator	2	1	0	1	1	1	1	1	1	1	1	0	0	1	1	1	0	0
ROVM	Reset overflow mode	1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	0	1	0
SOVM	Set overflow mode	1	1	0	1	1	1	1	1	1	1	1	0	0	0	1	0	1	1
SST	Store status register	1	1	0	1	1	1	1	1	0	0	I	←			D			→

I/O AND DATA MEMORY OPERATIONS

MNEMONIC	DESCRIPTION	NO. CYCLES	NO. WORDS	OPCODE INSTRUCTION REGISTER 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMOV	Copy contents of data memory location into next location	1	1	0	1	1	0	1	0	0	1	I	←			D			→
IN	Input data from port	2	1	0	1	0	0	0		PA		I	←			D			→
OUT	Output data to port	2	1	0	1	0	0	1		PA		I	←			D			→
TBLR	Table read from program memory to data RAM	3	1	0	1	1	0	0	1	1	1	I	←			D			→
TBLW	Table write from data RAM to program memory	3	1	0	1	1	1	1	1	0	1	I	←			D			→

APPENDIX (F)

SLAVE PROCESSOR BUFFER MODES

APPENDIX (F)

TMS32010 SLAVE PROCESSOR BUFFER MODES

To provide the most flexible operation as possible the RAM associated with a slave processor may be allocated for use in a variety of different configurations. The various modes listed below are designed so as not to inhibit slave processing.

MODE A (code 0) All buffers off, usually from either system or power up reset.

MODE B (code 1) Simple download, master access to slave program memory, all other buffers disabled.

MODE C (code 2) Slave access to program and scratchpad memory, window set for a dump from frame.

MODE D (code 3) Slave access to program and scratchpad memory, window set for a dump to frame.

MODE E (code 4) Slave access to program memory, master access to scratchpad, window set to dump from frame.

MODE F (code 5) Slave access to program memory, master access to scratchpad, window set to dump to frame.

MODE G (code 6) Slave access to program and window memory, master access to scratchpad memory.

MODE H (code 7) Slave has access to program, scratchpad and window memory.

MODE I (code 8) Master access to program and scratchpad memory, window disabled.

MODE J (code 9) Master access to program and scratchpad memory, window set for dump from frame.

MODE K (code A) Master access to program and scratchpad memory, window set for dump to frame.

APPENDIX (G)

VECTOR LINESCAN CONSTANTS

APPENDIX(G)

VECTOR LINE SCAN CONSTANTS

The Vector Orthogonal Scanner is a very powerful and flexible memory access tool. Due to its complexity,control is by no means trivial. However if suitably driven the machine can be made to scan out any pattern, curve or trajectory within a frame store. The major effort in driving the VOS is the calculation of the constant coefficients used by the machine to control the scanning action. In the extreme case such coefficients may fill a 1kx24 driver databank.

The method of operation of the VOS and its associated parameters were introduced in chapter 3. However it would seem worthwhile to include another practical example to further clarify the drive requirements of the machine. This takes the form of the coefficient calculation to produce the line scans for the vector scanning shape descriptor.

To practically realise the vector scanning shape descriptor requires that the VOS traces 32 scan lines through the target centroid. The test is currently carried out over a 89x89 grid with the target coincident with the centre (0,0). The important parameters to consider as far as scan production is concerned are shown overleaf.

PARAMETER	DESCRIPTION
XS	- X position of start of scan relative to centre (8 bit 2s complement value)
YS	- Y position of start of scan relative to centre (8 bit 2s complement value)
LINC	- line increment value
NLINC	- new line increment/decrement value
EOL	- number of points per line (minus 2). The offset is subtracted from the value due to a hardware constraint which requires EOL to be phase advanced.
LCNT	- number of double layers to be filled in databank stack prior to the ram based EOD flag being set.
EOD	- end of dump flag, terminates dump when it is found set in a databank stack layer.
HSCANFLAG	- flag which when set indicates negative jump mode on NLINCs. This invokes 1s complement arithmetic in order to avoid violating pseudo random store access.

The following tables show the values of driver parameters for given angular scans.

ANGLE	YS	XS	LINC	NLINC	EOL	LCNT	HSCANFLAG
45	A8	2B	02D0	086D	1	1E	0
50.6	A8	22	02D0	0E0C	3	12	0
56.3	A8	1D	02D0	086E	1	1E	0
61.9	A8	15	02D0	0B3E	2	16	0
67.5	A8	11	02D0	0E0E	3	12	0
73.1	A8	0C	02D0	1C1D	8	09	0
78.8	A8	08	02D0	0E0F	3	12	0
84.4	A8	04	02D0	1C1F	8	09	0
90.0	A8	00	02D0	0167	57	01	0
95.6	A8	FC	02D0	1C21	8	09	0
101.3	A8	F8	02D0	0E11	3	12	0
106.9	A8	F4	02D0	1C23	8	09	0
112.5	A8	EF	02D0	0E12	3	12	0
118.1	A8	EB	02D0	0B42	2	16	0
123.8	A8	E3	02D0	0872	1	1E	0
129.4	A8	DE	02D0	0E14	3	12	0
135.0	A8	D5	02D0	0873	1	1E	0
140.6	B8	D4	0001	070B	1	1E	0
146.3	C6	D4	0001	05A3	1	1E	0

ANGLE	YS	XS	LINC	NLINC	EOL	LCNT	HSCANFLAG
151.9	D5	D4	0001	043B	1	1E	0
157.5	DE	D4	0001	05A5	3	12	0
163.1	E7	D4	0001	043D	3	12	0
168.8	EF	D4	0001	02D5	3	12	0
174.4	F8	D4	0001	016D	3	12	0
180.0	00	D4	0001	0169	57	01	0
-174.4	08	D4	0001	1E9D	3	12	1
-168.8	11	D4	0001	1D35	3	12	1
-163.1	19	D4	0001	1BCD	3	12	1
-157.5	22	D4	0001	1A65	3	12	1
-151.9	2B	D4	0001	1BCB	1	1E	1
-146.3	3A	D4	0001	1AD4	1	1E	1
-140.6	48	D4	0001	1BFB	1	1E	1

N.B.

Angles are in degrees decimal, the remaining entries are in hexadecimal.

EOL has been predecremented.

Each scan allows 2 eventual radii measurements.

These values can be found in the program listings, notably RADCOR4. The coefficients are held in the table SCNTAB in compacted form as shown below.

TABLE ENTRY

WORD A WORD B WORD C WORD D

WORD A - YS7-YS0:XS7-XS0

WORD B - EOL3-EOL0:LINC11-LINC0

WORD C - EOL3-EOL0:NLINC11-NLINC0

WORD D - HSCANFLAG:LCNT6-0:NLINC12:EOD:EOL9-EOL4

N.B.

The numbers imply bit positions.

EOD is always 0 in the table.

APPENDIX (H)

STATE MACHINE DEVELOPMENT UNIT

SWITCHED MODES

APPENDIX (H)

THE STATE MACHINE DEVELOPMENT UNIT DEBUG MODES

The state machine development is a synchronous state machine using RAM based feedback logic and output decoders. The microcode is initially held in eprom and booted into the SMDU under software control. Once proper microcode is installed the hardware functions as a high speed timing sequencer with clock rates up to 28 MHz. In addition to its essential elements the SMDU offers a diagnostic feature to assist microcode development. This takes the form of a bar graph state display, single step facility and switches to allow changes of mode and the reading back of RAM contents.

The switching descriptions and options listed below are fairly complex and should be studied in conjunction with the circuit diagram (ref. 29).

SWITCH BANK E4

	ON	OFF
1	PULSECLK/2 SOURCE TO SMDU INPUT	DEFAULT NC
2	SSTEPCLK/2 SOURCE TO SMDU INPUT	DEFAULT NC
3	AUX LED ENABLE	" OFF
4	BARGRAPH ENABLE	" OFF
5	A12 EPROM NC OR DEFAULT LOW	
6	A11 EPROM NC OR DEFAULT LOW	
7	A10 EPROM SOFTWARE CONTROL OR DEFAULT LOW	
8	A9 EPROM SOFTWARE CONTROL OR DEFAULT LOW	
9	A8 EPROM SOFTWARE CONTROL OR DEFAULT LOW	
10	SMDU BOOT UNDER SOFTWARE CONTROL OR DEFAULT CONTINUOUS	

SWITCH BANK K6

	ON	OFF
1	CLKOUT TO COUNTER	NC
2	SSTPCLK TO COUNTER	NC
3	EN TO CLR	NC
4	CLR LOW	CLR HIGH
5	EN TO EN2	NC
6	SSTEP O/P	NC
7	$\overline{\text{EN}}$ TO $\overline{\text{EN2}}$	NC
8	SSTEP FB	NC
9	EN TO EN 1	NC DEFAULT LOW
10	$\overline{\text{EN}}$ TO $\overline{\text{EN 1}}$	NC DEFAULT HIGH

SWITCH BANK A4 DESCRIPTION

		ON	OFF
1	FREERUN PULSECLK TO SMDU		NC
2	SSTEP CLK TO SMDU		NC

NORMAL MODE OPERATION SWITCH SETTINGS

	E4	K6	A4
1	ON	ON	ON
2	OFF	OFF	OFF
3	ON	ON	
4	ON	OFF	
5	OFF	ON	
6	OFF	OFF	
7	OFF	ON	
8	OFF	OFF	
9	OFF	ON	
10	ON	ON	

SSTEP MODE OPERATION SWITCH SETTINGS

	E4	K6	A4
1	OFF	ON	OFF
2	ON	OFF	ON
3	ON	ON	
4	ON	OFF	
5	OFF	ON	
6	OFF	OFF	
7	OFF	ON	
8	OFF	OFF	
9	OFF	ON	
10	ON	ON	

In SSTEP mode the SMDU freerunning clock is replaced by the operator activating the toggle switch to generate pulses. Note that because of complex timing constraints the single step function may not operate the VOS in exactly the same way as the freerunning clock, so use with care!

RAM READ BACK SWITCH SETTINGS/USAGE

	E4	K6	A4
1	OFF	OFF	OFF
2	ON	ON	ON
3	ON	OFF	
4	ON	OFF COUNT CLR	) MANUAL
		ON COUNT ACTIVE	) COUNTER CLR

Switch K6(4) is used for resetting the RAM address count.

5	OFF	OFF
6	OFF	ON COUNT TO O/P RAMS

If K6(6) is off then O/P decoder RAMS have their addresses single stepped and data is displayed on the bargraph.

		OFF NC
7	OFF	OFF
8	OFF	ON FB RAM TO O/P RAMS

If K6(8) is off then feedback RAM has its addresses single stepped through and data is displayed on the bargraph.

		OFF NC
9	OFF	OFF
10	OFF	OFF

APPENDIX (I)

LISTS OF VOSTTAC 1 PROGRAMS, ROUTINES, TABLES & CONSTANTS

APPENDIX

GLOBALS

Name	Description
LS1ZRL	Real Line Size (360)
WS1ZRL	Window Capacity in Words (8K)
RSCE	Number of Available Slaves
BRANCH	Code for Branch Instruction
NLINRL	Number of Lines, Real (577)
LSIZGD	Grid Line Size
NLINGD	Grid Number of Lines
VREDUC	Vertical Conversion Factor (NLINRL/NLINGD)
HREDUC	Horizontal Conversion Factor (LSIZRL/LSIZGD)
STRLHI	Real Y Cord Start
STRLLO	Real X Cord Start
NLN	Jump to Newline Value (VOS)
EOL	End of Line Value (VOS)
LIN	Line Increment Value (VOS)
ERFLG1	Error Code
ERFLG2	Error Argument
BISTAT	BIO Status
WINNUM	Number of Lines which Fit in Window
STKPNT	Stack Pointer
RESNUM	Resolution Number

Name	Description
LINNUM	Number of Lines which fit in Window
DLBEGP	Dump List Start Pointer
DLENDP	Dump List End Pointer
RREDUC	Area Reduction Factor
RMEAN	Mean Radius
RPNT1	Result Pointer 1
RPNT2	Result Pointer 2
SCNPT1	Scratch Pad Pointer 1
SCPNT2	Scratch Pad Pointer 2
WPNTR1	Window Pointer 1
WLINES	Number of Lines in Window
WPIXES	Number of Pixels on Window Line
GMASK	Data Mask
CENX	Copy of STRLLO
CENY	Copy of STRLHI
ACCL	Temporary Accumulator Copy Location
ACCH	Temporary Accumulator Copy Location
WEIGHT	Weighting Value for RADii
FM	Square Root Coefficient
BM	Square Root Coefficient
AM	Square Root Coefficient
YM	Square Root Coefficient
INVRT2	(One over Root 2)
MNTSA	Floating Point Mantissa
EXPNTA	Floating Point Exponent

Name	Description
LNSKIP	Line Skip Factor
STRMLO	Absolute Start Address Low
STRMHl	Absolute Start Address High
HINC	Vertical Conversion Coefficient (LSIZGD/LSIZRL)
VINC	Horizontal Conversion Coefficient (NLINGD/NLINRL)
COMREG	Executive Register
TSKPNT	Task Pointer

N.B.

This is not an exhaustive list and other variables are used and often overlaid due to memory constrictions more information is available in the VOSTTACl TMS32010 listings manual.

Application Programs

Program	Description
RADCOR4.TMS	Vector Scanning Shape Description/Comparison
VSCANHOGH.TMS	Hough Transform on Vector Scanned Images
SEGMENT4.TMS	Blob Segmenter/Multi Target Tracker (On-Line)
HSTOGRM3.TMS	Multispectral Thresholding (software)
TESTSUB2.TMS	Movement Detector (On-Line)
PRDTRKPLL.TMS	Predictive Tracker (On-Line, Early Executive)
HARDEDGE5.TMS	Hardware Edge Detection (On-Line)
SCREENS1.TMS	BBC Model B Interface Drivers
SCREENS2.TMS	BBC Model B Interface Drivers
ZOOMEDGE.TMS	Hardware Zoom with Edge Detector
PATHFND2.TMS	Road Centre/Edge Tracker (On-Line)
CONVOLVER.TMS	3X3 Template Mask Convolver
TRACKERI5.TMS	Intensity Tracker (On-Line, New Executive)
NEWHST.TMS	Histogrammer/Thresholder
HARDHST6.TMS	Hardware Thresholding (On-Line)
PATHFNDER.TMS	Early Road Tracker
SYSIE.TMS	Early System Core
POSTERIZE.TMS	Artificial Resolution Reducer
QUANTIZE.TMS	Artificial Sample Range Reducer
TRACKERMT.TMS	Multispectral Colour Tracker (On-Line, New Executive)
VSCANEDGE.TMS	Diagonal Edge Detection
MINSYS2.TMS	Basic System Core
RANGING.TMS	Range Filters on 3x3 Window
RANKING.TMS	Rank Filters on 3x3 Window

These programs represent a small subsection of those available. A full list is given in the VOSTTAC1 TMS320 Listings File, along with printouts of most of the above.

List of Routines/Tables

Routine/Table	Description
COEF	Program to Datamemory Load Utility
SYBOOT	Slave Boot Up Program
SLRST1	Slave Reset Routine
SLINT1	Slave Vectored Interrupt Controller
SVINT1	Slave Interrupt Routine Partial Result Finder
SVINT2	Slave Interrupt Routine Vector Analyser
SVINT3	Slave Interrupt Routine Copies Internal to External Datamemory
SVINT4	Slave Interrupt Routine Vector Analyser
SVINT5	Slave Interrupt Routine Transfer Window to Scratchpad
SVINT6	Slave Interrupt Routine Transfer Scratchpad to Window
SVINT7	Slave Interrupt Routine Write to BBC B Interface
SVINT8	Slave Interrupt Routine Read from BBCB Interface
SVINT9	Slave Interrupt Routine Histogram Computer
SVINT10	Slave Interrupt Routine Blob Segment
XNTORL	Converts Grid to Real X
VEC320	Gives Slave Vectored Interrupt
MNBOOT	Performs Single Slave Boot
RSHF32	32 Bit Right Shift of 12
GETWIN	Gets Screen Window into Scratchpad For Master Access
MOVWIN	Copies Slave Window to Scratchpad

Routine/Table	Description
MOVSCP	Copies Slave Scratchpad to Window
YNTORL	Converts Grid Y to Real Y
NSTORL	Converts Grid Start to Real Start
XYTOAD	Converts Real X, Y to Absolute Address
LUNAK	Returns Port Address from LUN
DIVAN	General Division Routine
DRIVAD	Fixed Boundary VOS Dump/Loader
VECZ80	Gives Vectored Interupt to NOM.RIG
BRQZ80	Request Nom Rig Bus
SMDUBT	Boots Up VOS Timing Sequencer
VIDEOZ	Initialises/Controls Video Section
TOGON	Software Synchronization to Frame Timing
NMVID	Video Constants Set Up
GETBLK	Reads 64 words from Scratchpad
FILBLK	Writes 64 Words to Scratchpad
SMEMRW	General Slave to Master Block Transfer
FSTNOM	Target/Area Nominate
VIDEO	(Early Version of VIDEOZ)
FNDMID	Converts Window Co-Ordinates to Real Ones
CROSS	Drives Nomination Cross on Nom.Rig
CRSTST	Tests Cross
ZHLTOK	Ensures Z80 In Halt State
SETFAC	Sets up Various Constants
SEEREG	Lets Master Read Slave Internal Data Memory
READIT	(Used in Conjunction with SEEREG)
RSTOFF	Removes Slave Reset

Routine/Table	Description
DMYDMP	Dummy Test of VOS
ADDRST	Clears VOS BIO Line
CLMBIO	Clears Slave MBIO Line
RSHF9	Right Shift 9
RSHF12	Right Shift 12
RSHF2	Right Shift 2
RSHF6	Right Shift 6
WTFBIO	Waits for BIO to activate
PALTTE	Colour Palette
TEMP3	Simple Convolver
TVALS	Sets Template for TEMP3
CLRWIN	Clears Scratchpad
DRAXIS	Draws Axis in Scratchpad
CORDSC	Scales Waveform to Fit on Axis
DWIN	Draws Waveform in Scratchpad
SETPNT	(Pointer Reset)
GRPOUT	Displays Scratch Pad on Screen with Variable Position & Zoom
ZOOM	Variable X and Y Stretch Zoom
TABBIO	Table of Addresses for BIO Levels
SLVTAB	Table of Addresses for Available Slaves
SLBTAB	Table of BIOs for Available Slaves
LKUPTB	Table of Set Bit Positions
BISERV	Addresses of BIO Service Routines
SCNTB	Constants for Line Vector Scanning
WGHTTB	Angular Weighting Coefficients
SVECTB	Slave Interrupt Service Addresses

Routine/Table	Description
SLCOF1	Slave Initialisation Constants
RESTAB	Resolution Skipping Coefficients
TSKTAB	Task Address Table (Early Executive Only)
SBTB1	Subtask Address Table (Early Executive Only)
DMPSTK	Dump Queue Table (Early Executive Only)
WKSPCE	Program Memory Work Space Area
TASKON	Task Table Controller (Early Executive Only)
STKPSH	Int Datamemory Stach Pusher
STKPOP	Int Datamemory Stack Popper
ERTRAP	Error Detector/Reporter Utility
WHOSIT	BIO Source Identifier
SETCOM	Sets COMREG Bit by LSLN
SETBIT	Sets COMREG Bit by BIO
RESCOF	Finds Required Pixel and Line Skip Values
ARATIO	Computes Resolution Reduction Factor
FNSTRT	Finds Start Addresses for Multiple Dumps
CENBOX	Centres Next Dump Around Target
ARMTCH	Calculates % Area Match with Target
WKSTRE	Stores Value in Workspace
WKREAD	Reads Value in Workspace
TRSHER	Software Double Edged Tresholds on RGB Space
INITST	Reads Slave Program Memory
BOOTEM	Boots Up All Available Slaves
RSCFND	Finds All Available Slaves
SLMODE	Sets slave Buffer Mode
UCALD	Subsection of Executive (Modern)

Routine/Table	Description
CLRBIT	Clears Bit in COMREG
DRIBOX	Handles Cross Driving and Window Tracking
GETBIT	Returns BIO Mask From LSLN
PREDIC	Predicts Target Position from Recent History
TOGBIO	Clears Video BIO
TGTIDI	Target Finder 1 Slave Sequence
TGT1D2	Target Finder 2 Slave Sequence
CHAROM	Mini Character ROM
STRING	String Work Space
CHAREX	Expands Character Words Into DDM
ALPHA	Modifies String
PKSTRG	Packs a String
DECOUT	Outputs Decimal Values
APEX	Finds Peak Percentage
SKEW	Converts Skew to Text
CLS	Clears Screen
HSTWIN	Gets 90x90 or 14x14 Window
HSTPKS	Finds Histogram Peaks
HP2LPF	Low Pass Filter on Histogram
HDISP	Displays Histogram
HISTOG	Gets RG&B Histograms
HDWIN	Used with HDISP
SCRMOD	General Full Screen Modifier Utility
TLIM5	Software Tresholder
TTAB5	5 Wide Peak Treshold
TTAB3	3 Wide Peak Treshold

Routine/Table	Description
GETVSC	Gets Vector Scan Line
VCINIT	Set Up For Vector Scan and Mode
VCTB	VOS Address Table
VSCDAT	Sector for Individual Vector Scans
VSCAN	VOS Line Scan Driver
FLTNRM	Fixed to Floating Point Normalisation
FLTMPY	Floating Point Multiply
FLTDIV	Floating Point Divide
ALIGN	Floating Point Alignment
FLTADD	Floating Point Add
FLTSUB	Floating Point Sub
FLTINV	Floating Point Inverse
FLTSQR	Floating Point Square Root
SQCNST	Sets Constants for Square Root
RAVGE	Finds Average RADII Measure
DCOFFR	Removes Average RADll From Measurements
CYCCOR	Performs Cyclic Correlation
BLKSVE	Stacks Data Mem in Program Memory
BLKLD	Stacks Prog Mem in Data Memory
NRMEAS	Measures RADII
RSTRE	Stores RADII in Lower Scratchpad
RDNRM	Weights RADII According to Angle
SCLOR	Calculates Fixed Point RADII Scale Factor
SIGSTR	Stores Scaled Function in Scratchpad
SCALEB	Scales RADII Relative to Peak of Autocorrelation Function
CORPOW	Integrates Squares of Correlation Function

Routine/Table	Description
BGSRCH	Finds Best Model Match
PEKABO	Finds Peaks in Correlation Waveform
CORING	Gets Info about Correlation
PRINT	Outputs Info in Textual Format
WINOUT	Puts Characters in Window
CONVER	Convolves 3x3 Template with Picture
QUANT	Artificially Quantizes Image
CNVSET	Sets up 3x3 Template Coefficients
POSTX	Artificial X Posterizer
POSTY	Artificial Y Posterizer
DUMPER	Performs Dumps (Early Systems)
DMPSEC	Interacts with Dump Queue (Early Systems)
QUEDMP	Handles Dump Queue (Early Systems)
DUMPIT	Used with Above (Early Systems)
VCBLOB	Blob Segmentor
PX4SM	4 Pixel Summator
PX16SM	16 Pixel Summator
PRIGAS	Primary Blob Group Assignment
SEGAS	Secondary Blob Group Assignment
FNDGPA	Blob List Reader
SORTA	Blob List Sorter
INTDIV	Integer Division Utility
GETCRD	Gets Co-ordinate of Specified Target
CRDCNV	Converts Co-Ordinate of Specified Target to Grid
DAC2XY	Outputs Blob Positions on DACS

Routine/Table	Description
PRIBIO	Main Executive Routine
GRIDIT	Performs 3x3 Multiply Adds
OUTYGO	Output Value Conditioner Prior to Display
MASKTB	Table of Convolution MASKS
HOUGH	Hough Transform Routine
BLKFIL	Erases Perimeter of Screen
SETSRC	Searches Window for Set Pixels
ARRAY	Modifies Hough Accumulator Array
MAXFIND	Finds Maximum Value in Array
DRWLNE	Draws Line in Window
TRACER	Outputs Tracking Values on DAC2

Note:

A fuller description of the preceeding routines/tables can be found in the VOSTTAC1 TMS32010 Listings File.

APPENDIX (J)

A SOFTWARE CHANNEL COMMUNICATIONS PROCEDURE

APPENDIX (J)

INTERSLAVE COMMUNICATIONS METHOD

A mechanism by which slaves could communicate with each other during rather than at the end of their subtasks might proceed as follows.

Each slave could have two 4 x 8 word buffer blocks, one each for send and receive. Each 8 word packet would consist of a source and destination address with 6 words of data. LSNs (Logical Slave Numbers) would be used for addressing with 16 being an easily tested case implying block free. Each slave could use variables TXCNT and RXCNT to maintain their buffers along with a flag (FLGCOM). FLGCOM would indicate to the master service routine whether the BIO generated, signals an end of subtask or a communications request.

The slaves would handle their buffer as follows:

Receiving

a) RXCNT≠0 to i.e. buffer not empty then point to start of receive buffer ELSE go to a

b) Test source if not required then go to C else D

c) Increment pointer by 8: If pointer too large then go to a else b

d) Source found so read message into program: decrement RXCNT: Clear block by changing source code: RETURN

Transmitting

a) If TXCNT≠4 then point to start of send buffer go to b ELSE a)

b) If source ≠ free then C ELSE D

c) Increment pointer by 8: If pointer too large go to a ELSE b

d) Set source to slavenumber: set destination: write message: Increment TXCNT: Set FLGCOM=1: Give MBIO: Return

In addition a slave should periodically examine FLGCOM and if set generate a MBIO. This is because the set flag represents an unsatisfied transmission request. To effect the actual data transfer requires that the master work quite hard to implement the communications channel. Having identified a BIO from a specific slave the master proceeds with the sequence.

1) BIO serving routine puts TX slave in halt state with full context save

2) If FLGCOM = 0 then clears MBIO: RETURNS (Normal Mode) ELSE 3)

3) If TXCNT $\neq$ 0 then 4 else 12

4) Read first message into master buffer

5) Close stream of TX but maintain halt

6) Read destination address and open stream

7) Halt RX slave with full context save

8) If RXCNT $\neq$ 4 then find free block: write message: increment RXCNT

ELSE Release RX halt, close stream: open TX stream release halt:

Clear MBIO: return

9) Release RX halt: close stream

10) Open TX stream: Decrement TXCNT: Clear Block

11) If TXCNT $\neq$ 0 then 4 ELSE 12

12) Set FLGCOM=0: Clear MBIO: Release Halt: RETURN

If exiting at 12 all transfers satisfied. If at 8 receiver buffer full so transmit is frustrated and FLGOM is left set to indicate a pending transmission.

APPENDIX (K)

CONVOLUTION TEMPLATES

APPENDIX (K)

CONVOLUTION TEMPLATES

RANK 3

a	b	c
d	e	f
g	h	i

NAME		COEFFICIENTS								
		a	b	c	d	e	f	g	h	i
KIRSCH/PREWITT	(V)	-1	0	1	-1	0	1	-1	0	1
" "	(H)	-1	-1	-1	0	0	0	1	1	1
SOBEL	(V)	-1	0	1	-2	0	2	-1	0	1
SOBEL	(H)	-1	-2	-1	0	0	0	+1	2	1
GRADIENT	(N)	1	1	1	1	-2	1	-1	-1	-1
"	(S)	-1	-1	-1	1	-2	1	1	1	1
"	(E)	-1	1	1	-1	-2	1	-1	1	1
"	(W)	1	1	1	1	-2	-1	1	1	-1
"	(NE)	1	1	1	-1	-2	1	-1	-1	1
"	(NW)	1	1	1	1	-2	-1	1	-1	-1
"	(SE)	-1	-1	1	-1	-2	1	1	1	1
"	(SW)	1	-1	-1	1	-2	-1	1	1	1
LPF	(1)	1/16	1/8	1/16	1/8	1/4	1/8	1/16	1/8	1/16
"	(2)	1/10	1/10	1/10	1/10	1/5	1/10	1/10	1/10	1/10
"	(3)	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9	1/9
HPF	(1)	-1	-1	-1	-1	9	-1	-1	-1	-1
"	(2)	0	-1	0	-1	5	-1	0	-1	0
"	(3)	1	-2	1	-2	5	-2	1	-2	1
LAPLACIAN	(1)	-1	-1	-1	-1	8	-1	-1	-1	-1
"	(2)	0	-1	0	-1	4	-1	0	-1	0
"	(3)	1	-2	1	-2	4	-2	1	-2	1
LINESEG	(V)	-1	2	-1	-1	2	-1	-1	2	-1
LINESEG	(H)	-1	-1	-1	2	2	2	-1	-1	-1
LINESEG	(45°)	-1	-1	2	-1	2	-1	2	-1	-1
LINESEG	(-45°)	2	-1	-1	-1	2	-1	-1	-1	2

ACKNOWLEDGEMENTS

Thanks are due primarily to Bath University and Honeywell Aerospace and Defence U.K. (formerly Honeywell Leafield Ltd.) for providing the facilities and funding which made this research possible.

As far as credit to individuals is concerned, my academic supervisor Ken Maccormac must top the list. It was his endeavour that resulted in the set up of the industrial collaboration and his decision to allow myself to carry out the research. In addition, from his successful background in industry, he has acquired project management skills which played a major part in keeping to the ambitious schedule of work and focused on realistic objectives. For these reasons and others to numerous to mention, a big vote of appreciation to Ken.

High up on the list of individuals to be credited must be Alan Tyler of Honeywell, who was until recently the main contact with the University. Alan's enthusiasm for the work done at Bath and his ready assistance was an invaluable asset to the research effort. More recently Alan has helped provide financial and practical assistance for the speedy completion of this document, for which I owe a great debt of gratitude.

Recognition must also be given to two "conscript" thesis assistants, namely Annette Dobson (yes it really is finished !) and Lionel Chandler who have both contributed to the physical preparation of this publication.

To conclude, I must issue a blanket thankyou to everyone who has assisited in this research effort whether at University, Honeywell or the world at large.

REFERENCES

1) Marr D., 1982, "Vision", W.H.Freeman & Co San Francisco

2) Hartline H.K., 1938, "The Response of Single Optic Nerve Fibres of the Vertebrate Eye to Illumination of the Retina", Amer.J.Physiol 121,400-415

3) Barlow H.B., 1964, "Retinal Ganglion Cells Responding Selectively To Direction and Speed of Image Motion in the Rabbit", J.Physiol 173,377-407

4) Barlow H.B., 1953, "Summation and Inhibition in the Frog's Retina", J.Physiol 119,69-88

5) Paul G., 1979, "Large Scale Vector/Array Processors", Advances in Image Processing (P.Stucki,ED.) Plenum Press, NY, 265-276

6) Unger S.H., 1958 "A Computer Oriented Towards Spatial Problems", Proc.Ire. 46,1744-1750

7) Duff M.J.B., 1981, "The Elements of Digital Picture Processing", real-time parallel computing (Onoe-Preston-Rosenfeld), Plenum Press, NY, 1-9

8) Duff M.J.B., 1981, "Review of the Clip Image Processing System", Proc. National Computer Conference.

9) Gerritsen F.A., 1983, "A Comparison of the CLIP4, DAP and MPP Processor Array Implementations", Computing Structures for Image Processing, Academic Press, 15-29

10) "CLIP", 1985, Stonefield Vision Systems

11) Fountain T.J., 1983, "A Survey of Bit-Serial Array Processor Circuits", Computing Structures for Image Processing, Academic Press, 1-13

12) Batcher K.E., 1980, "Architecture of a Massively Parallel Processor", IEEE Trans on Computing C29,836-840

13) Reddaway S.F., 1973, "DAP - A Distributed Array Processor", First Anual Symposium on Computer Architecture, Florida, 61-65

14) Ellis T.J., Styles D.E., 1985, "High Speed FFT Processor for Microcomputers", Fourth Symposium on Microcomputer and Microprocessor Applications, 38-51

15) "TMS32010 User's Guide", 1983, Texas Instruments

16) "DSP128 Data Sheet", 1984, S.T.C. Components

17) "AMI S7720 Data Sheet", 1984, A.M.I. Semiconductors

18) "MB8764 DSP Data Sheet", FUJITSU

19) "UDP101 DSP Data Sheet"

20) McLean T.P., 1985, "Keynote Address", Unmanned Land Vehicle Symposium to D.M.A. at the Royal Overseas League, London

21) Mayes K.E., Maccormac K., 1985, "The Design of a Vector Scanning Pattern Recognition System within a TMS32010 Multiprocessor Architecture", Fourth Symposium on Microcomputer and Microprocessor Applications, Budapest, 17-26

22) Mayes K.E., Maccormac K., 1986, "VOSTTAC1, The Vector Orthogonal Scanning, Target Tracking, Adaptive Controller", IEE Second International Conference on Image Processing and It's Applications, London, 171-174

23) Smol G., 1981, "Telecommumications, A Systems Approach", George Allen and Unwin, London

24) Rosenfeld A., KAK A., 1982, "Digital Picture Processing by Computer", Academic Press

25) Castleman K., 1979, "Digital Image Processing", Prentice Hall, New Jersey

26) Ballard H., Brown C., 1982, "Computer Vision", Prentice Hall

27) "Digital Coding Standards "IBA Technical Review, Volume 16

28) Chapman D.M., 1985, "Target Nomination and Video Signal Digitising for an Intelligent Tracking System", Final Year Project Report for BSC in Elec/Eng, Bath University

29) Mayes K.E., 1984-1986, "VOSTTACl Circuit Diagram File", (Internal Document)

30) "TMS32010 Evaluation Module, Manual", 1983, Texas Instruments

31) Haralick R.M., 1981, "Some Neighborhood Operators", Real-Time Parallel Computing (Onoe-Preston-Rosenfeld), Plenum Press, N.Y., 11-35

32) Nagin P.A. Et Al, 1981, "Region Relaxation in a Parallel Herachical Architecture", Real-Time Parallel Computing, (Onoe-Preston-Rosefeld), Plenum Press, N.Y., 37-61

33) Miller R., Stout Q., 1984, "The Pyramid Computer for Image Processing, IEEE Proc. 7th Patt.Recog.Conf., 240

34) Cantoni V., Levialdi S., 1982, "Matching the Task to an Image Processing Architecture", IEEE Proc 6th Patt.Recog.Conf, 254-257

35) Prescott N.G., 1984, "Report on the Development of an Interface Module for a Microprocessor Based System for Use in System Identification", Final Year Project Report for BSC in Elec/Eng, Bath University

36) Marr D., Hildreth E., 1979, "Theory of Edge Detection", MIT Artificial Intelligence Laboratories Report No AI-M-518

37) "The TTL Data Book (Vol 1&2)", 1985, Texas Instruments

38) "MECL Device Data", Motorola Inc.

39) "The Bipolar Microcomputer Components Data Book for Design Engineers", 1981, Texas Instruments

40) "AMD Programmable Array Logic Handbook", 1984, Advanced Micro Devices

41) "AMD BIPOLAR/MOS Memories Data Book", 1984, Advanced Micro Devices.

42) CATT.I., 1979, "Digital Hardware Design", Macmillan Press Ltd, London

43) Mayes K.E., 1985, "VOSTTAC1 I/O File", (Internal Document)

44) Carr J.,1980,"Z80 Users Manual",Reston;VA

45) Mayes K.E., 1986, "VOSTTAC1 Program Source Listings File", (Internal Document)

46) Cody.W, Waite W., 1980 "Software Manual for the Elementary Functions", Englewood Cliffs N.J., Prentice Hall, Series in Computational Mathematics.

47) Weszka, 1978, "A Survey of Threshold Selection Techniques", Computer Graphics and Image Processing 7,259-265

48) Wall R.J., 1974, "Analysis of Image Histograms", Proc 2nd Int.Conf. on Patt.Recog, Copenhagen, 341-344

49) Nevatia, Ramakant, 1982, "Machine Perception", Englewood Cliffs N.J.; Prentice Hall

50) Triendl E.E., 1978, "How to get the Edge into the Map" Proc 4th Int. Conf. on Patt. Recog., 946-950

51) Ohlander R. Et Al, 1978, "Picture Segmentation using a Recursive Splitting Method", Computer Graphics and Image Processing 8, 313-333

52) Horowitz S.L. Pavlidis T., 1974, "Picture Segmentation by a Directed Split and Merge Procedure", Proc., 2nd IJCPR, 424-433

53) Zucker S.W., 1976, "Region Growing : Childhood and Adolescence", Computer Graphics and Image Processing 5, 382-399

54) Lutton S.M., Mitchel O.R., "1980", Adaptive Segmentation of Unique Objects", Proc 5th Int. Conf. on Patt. Recog., 548-550

55) Gupta J.N., Wintz P.A., 1975, "A Boundary Finding Algorithm and Its Application", IEEE Trans. on Circuits and Systems, Vol. CAS-22, No 4, 351-362

56) Rosenfeld A., Davis L.S., 1979, "Image Segmentation and Image Models", Proc. IEEE, Vol. 67, No 5, 764-772

57) Davis L.S., 1975, "A Survey of Edge Detection Techniques", Computer Graphics and Image Processing 4, 248-270

58) Nevatia R., Babu K.R., 1980, "Linear Feature Extraction and Description", Computer Graphics and Image Processing 13,257-269

59) FU K.S., Mui J.K., 1981, "A Survey on Image Segmentation", Pattern Recognition, Vol. 13, 3-16

60) Heuckel M.H., 1973, "A Local Visual Operator Which Recognizes Edges and Lines", JACM, Vol.20, No 4,634-647

61) Heuckel M.J., 1971 "An Operator Which Locates Edges in Digitized Pictures", JACM, Vol. 18, No 1, 113-125

62) Mero L., Vassy Z., 1975, "A Simplified and Fast Version of the Heuckel Operator for Finding Optimal Edges in Pictures", Proc 4th IJCAI, 650-655

63) Georgiou C.J., 1986, "Architecture for a Single Chip Performing Real-Time Image Convolution", IEE 2nd International Conference on Image Processing and its Applications", 107-111

64) Abdou I.E., Pratt W.K., 1979, "Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors", Proc IEEE Vol. 67, No 5

65) Samwell C.J., Cain G.A., 1986, "The BAe (Bracknell) Automatic Detection, Tracking and Classification System", IEE 2nd International Conference on Image Processing and its Applications", 164-170

66) Cussons S., 1981, "Automatic Acquisition of Aircraft Targets using IR Imaging Sensors", IEE Conference on Advanced Infrared Detectors and Systems, 204,131-138

67) Nagao M., Matsuyama T., 1979, "Edge Preserving Smoothing", Computer Graphics and Image Processing 9,394-407

68) Chin R.T., Chia-Lung, 1983, "Quantitative Evaluation of Some Edge-Preserving, Noise-Smoothing Techniques", Computer Vision, Graphics and Image Processing 23,67-91

69) Justusson B., 1978, "Noise Reduction by Median Filtering", Proc. 4th Int. Conf. on Patt Recog. 502-504

70) Huang T.S. Et Al, 1979, "A Fast Two-Dimensional Median Filtering Algorithm", IEEE Trans A SSP, Vol.27, No 1, 13-18

71) Rosenfeld A., 1970, "A Nonlinear Edge Detection Technique", Proc IEEE 58, 814-816

72) Hodgson Et Al, 1985, "Properties, Implementations and Applications of Rank Filters", Image and Vision Computing, Vol. 3, No 1, 3-14

73) Hodgson R.M., Bailey D.G., "Range Filters : Local-Intensity Subrange Filters and their Properties", Image and Vision Computing, Vol. 3, No 3, 99-110

74) Garibotto G., Lambarglli L., 1978, "Fast On-Line Implementation of Two-Dimensional Median Filtering", Electronics letters, Vol. 15, No 1, 24-25

75) Suciu R.E., Reeves A.P., 1982, "A Comparison of Differential and Moment Based Edge Detectors", Proc Conf. on Patt.Recog. and Image Processing, 97-102

76) Dubery J.S., 1986, "Investigation into Detection of Road Edges By Image Processing and Computer Image Generation", Final Year Project Report For BSC in Elec/Eng, Bath University

77) Nevatia R., 1976, "A Color Edge Detector", Proc 3rd IJC on Patt.Recog., Colarado Calif., 829-832

78) Hough P.V.C., 1962, "Method and Means for Recognizing Complex Patterns", United States Patent 3,069,654

79) Duda R.O., Hart P.E., 1972, "Use of the Hough Transformation to Dectect Lines and Curves in Pictures", Comm.ACM, Vol. 15, No 1, 11-15

80) Frei W., Chen C., 1977, "Fast Boundary Detection: A Generalization and a New Algorithm", IEEE Trans on Computers Vol. C-26, No 10, 988-998

81) Freeman H., 1961, "On the Encoding of Arbitrary Geometric Configurations", IRE Transactions on Electronic Computers EC-10, 260-268

82) Pavlidis T., 1978, "A Review of Algorithms for Shape Analysis", Computer Graphics and Image Processing 7, 243-258

83) Freeman H., 1961 , "On the Encoding of Arbitrary Geometric Configurations", IRE Transactions on Electronic Computers

84) Cussons S., 1982, "A Real-Time Operator for the Segmentation of Blobs in Imaging Sensors", IEE Electronic Image Processing Conference, Proc, 214,51-57

85) Gonzalez, Wintz, 1977, "Digital Image Processing", Reading MA: Addison-Wesley

86) Freeman H., 1974, "Computer Processing of Line Drawing Images", Computing Surveys, Vol. 6, No 1, 57-97

87) Hu M.K., 1962, "Visual Pattern Recognition by Moment Invariants", IRE Transactions on Information Theory, IT-8, 179-187

88) Dudani S.A., 1977, "Aircraft Identification by Moment Invariants", IEE Trans. on Computing, Vol. C26, No 1, 39-45

89) Mitchell O.R., Lutton S.M., 1978, "Segmentation and Classification of Targets in Flir Imaginery", SPIE Image Understanding Systems and Industrial Applications 155, 83-90

90) Roberts L.G., 1965, "Machine Perception of Three-Dimensional Solids", Optical and Electro-Optical Information Processing (EDS Tippett, Berkowitz, Clapp, Koester, Vanderburgh) MIT., 159-197

91) FriedBerg S.A., Brown C.M., 1984, "Finding Axes of Skewed Symmetry", IEEE Proc 7th Patt.Recog.Conf., 322-325

92) Gindi G.R., Gmitro A.F., 1984, "Optical Feature Extraction Via the Radon Transform", IEEE Proc. 7th Patt.Recog.Conf. 702-704

93) Moravec H.P., 1977, "Towards Automatic Visual Obstacle Avoidance", International Joint Conference on Artificial Intelligence, 584

94) Dickmanns E.D., Zapp A., 1985, "Guiding Land Vehicles Along Roadways by Computer Vision", congres Automatique (Afcet)233-243

95) Harrison S.J., 1986, "Character Learning and Recognition Using a Multiprocessing Computer System", Final Year Project Report for BSC in Elec/Eng, Bath University

96) Hodgson R.M., McNeill S.J., 1986, "The Design of Image Processing Systems for Real-Time Inspection Applications", IEE 2nd International Conference on Image Processing and its Applications, 126-129

97) "Transputer Information Pack", 1985, Inmos Ltd

98) "TMS32020 User's Guide", 1986 , Texas Instruments

99) Barnes J.G.P., "Programming in ADA", International Computer Science Series, Addison Wesley

100) Rogers M.W. (ED), 1984, "ADA: Language Compilers and Bibliography", ADA Companion Series, Cambridge University Press.

101) "VRTX: Versatile Real Time Executive Product Brief", 1986, Hunter and Ready Inc.

Table 1 *Picture and signal standards for the principal monochrome television systems*

	British 405-line system	*American 525-line system*	*British 625-line system*	*European CCIR-625 line system*	*French 819-line system*
Number of lines per picture	405	525	625	625	819
Field frequency (Hz)	50	60	50	50	50
Picture frequency (Hz)	25	30	25	25	25
Aspect ratio	4/3	4/3	4/3	4/3	4/3
Line frequency (Hz)	10125	15750	15625	15625	20475
Line period (μs)	98·8	63·5	64	64	48·84
Active field factor a_1 (see Section 5.6)	0·931	0·923	0·922	0·922	0·919
Active line factor a_2 (see Section 5.6)	0·814	0·826	0·812	0·812	0·805
Number of active picture lines	377	485	575	575	753
Video bandwidth (MHz)	3	4·2	5·5	5	10
Number of picture elements which can be resolved along a line drawn vertically down the screen (Kell factor = 0·7)	264	340	402	402	527
Number of picture elements which can be resolved along a horizontal line	483	422	572	520	786
Transmitted gamma (approximate)	2 to 2·5	2·2	2·2	2·2	1·7
Sense of modulation	positive	negative	negative	negative	positive
Black level as % of peak carrier	35	75	77	75	25
Blanking level as % of peak carrier	30	75	77	75	25
Peak white level as % of peak carrier	100	15	20	10	100
Bandwidth of r.f. transmission channel (MHz)	5	6	8	7	14
Attenuated (vestigial) sideband	upper	lower	lower	lower	upper
Bandwidth of Nyquist flank region on either side of video carrier (MHz)	0·75	0·75	1·25	1·25	2
Position of sound carrier with respect to vision carrier (MHz)	−3	+4·5	+6	+5·5	−11·15
Sound modulation	a.m.	f.m.	f.m.	f.m.	a.m.
Sound carrier deviation (kHz)		±25	±50	±50	
Sound pre-emphasis (μs)		75	50	50	

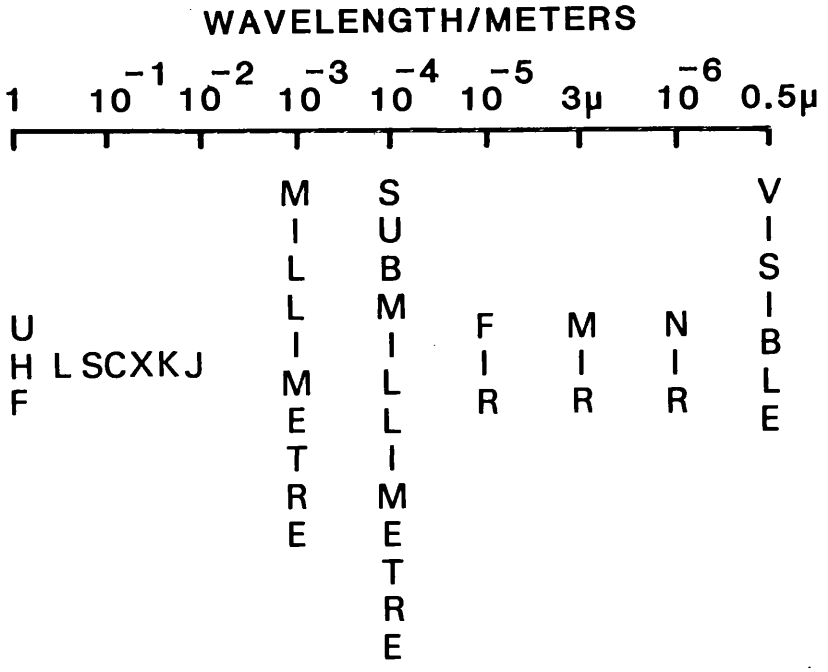


FIGURE 1 THE ELECTROMAGNETIC SPECTRUM

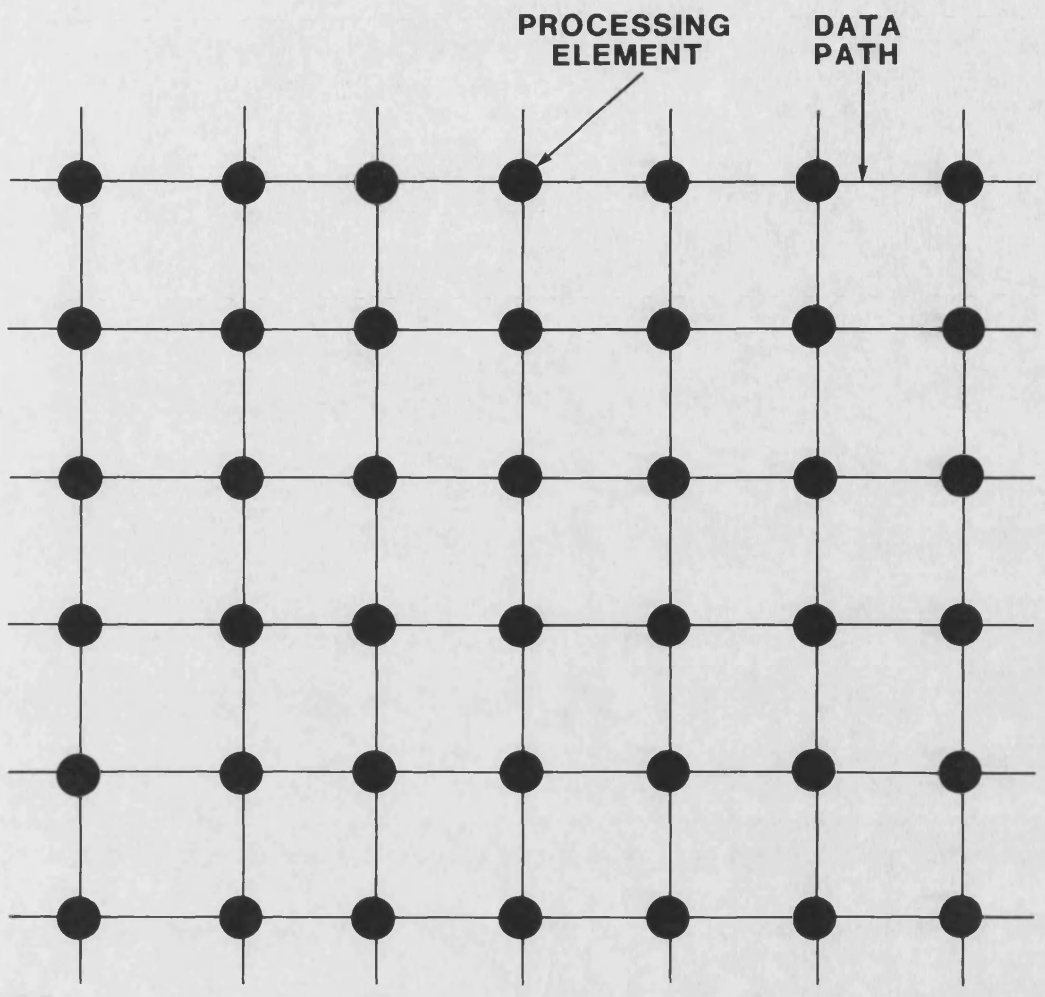


FIGURE 2 THE SIMD ARRAY

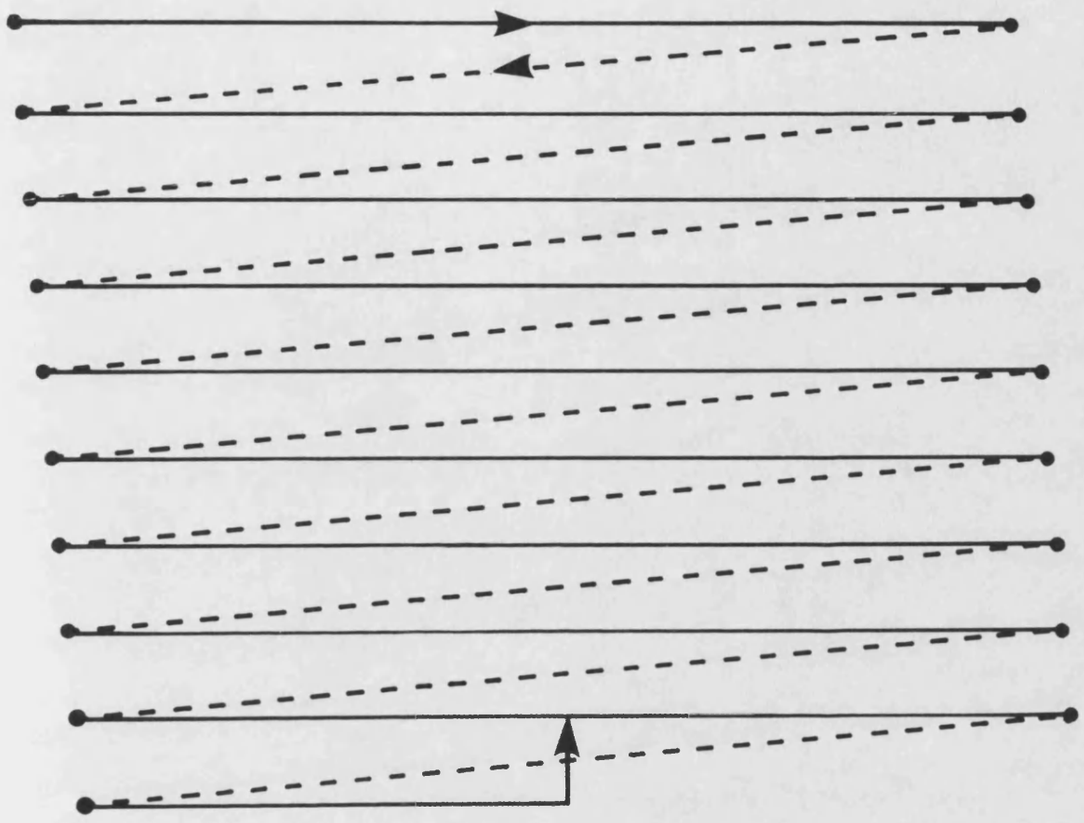


FIGURE 3 THE RASTER SCAN

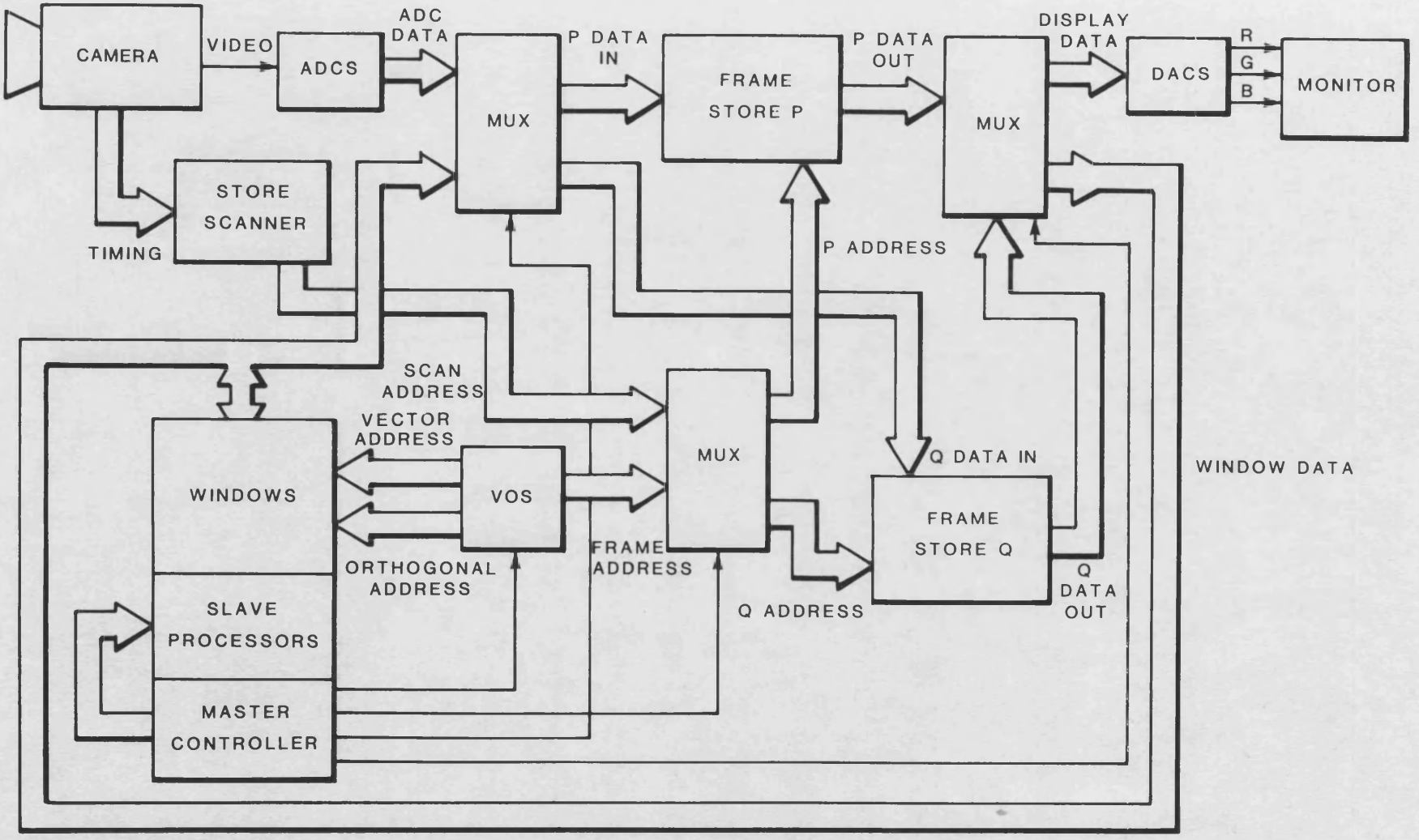


FIGURE 4 VOSTTAC 1 SYSTEM BLOCK DIAGRAM

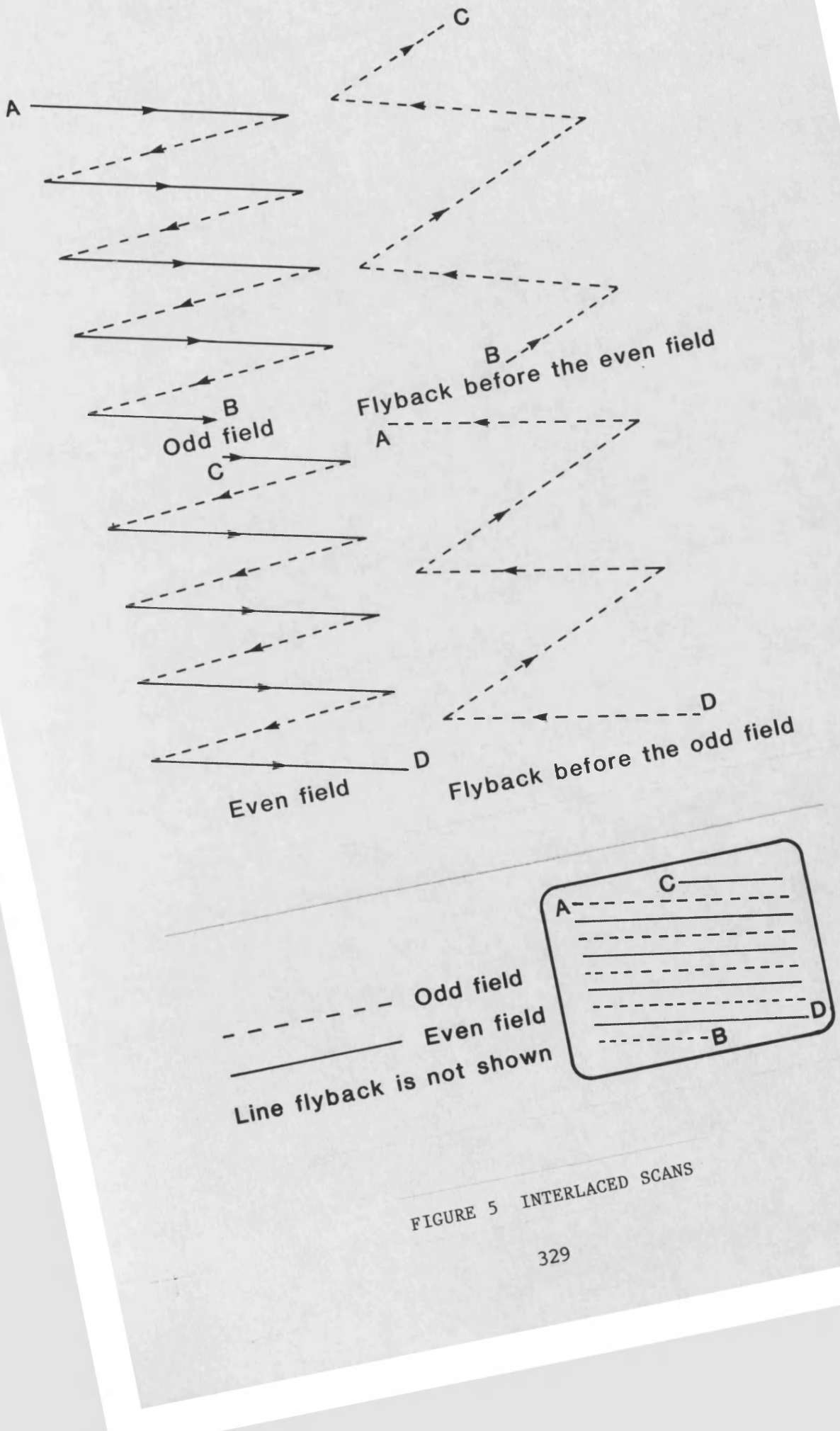


FIGURE 5 INTERLACED SCANS

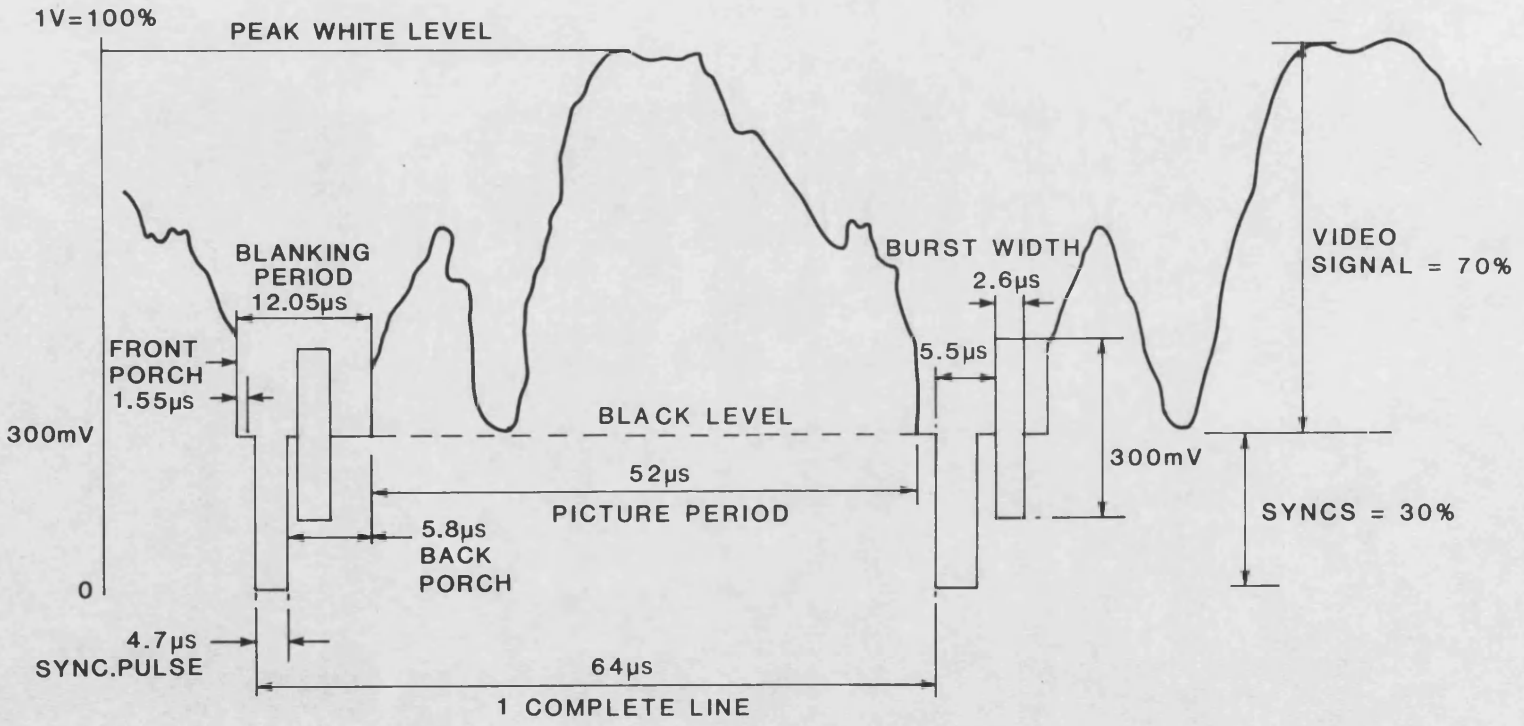


FIGURE 6 THE PAL 625 LINE

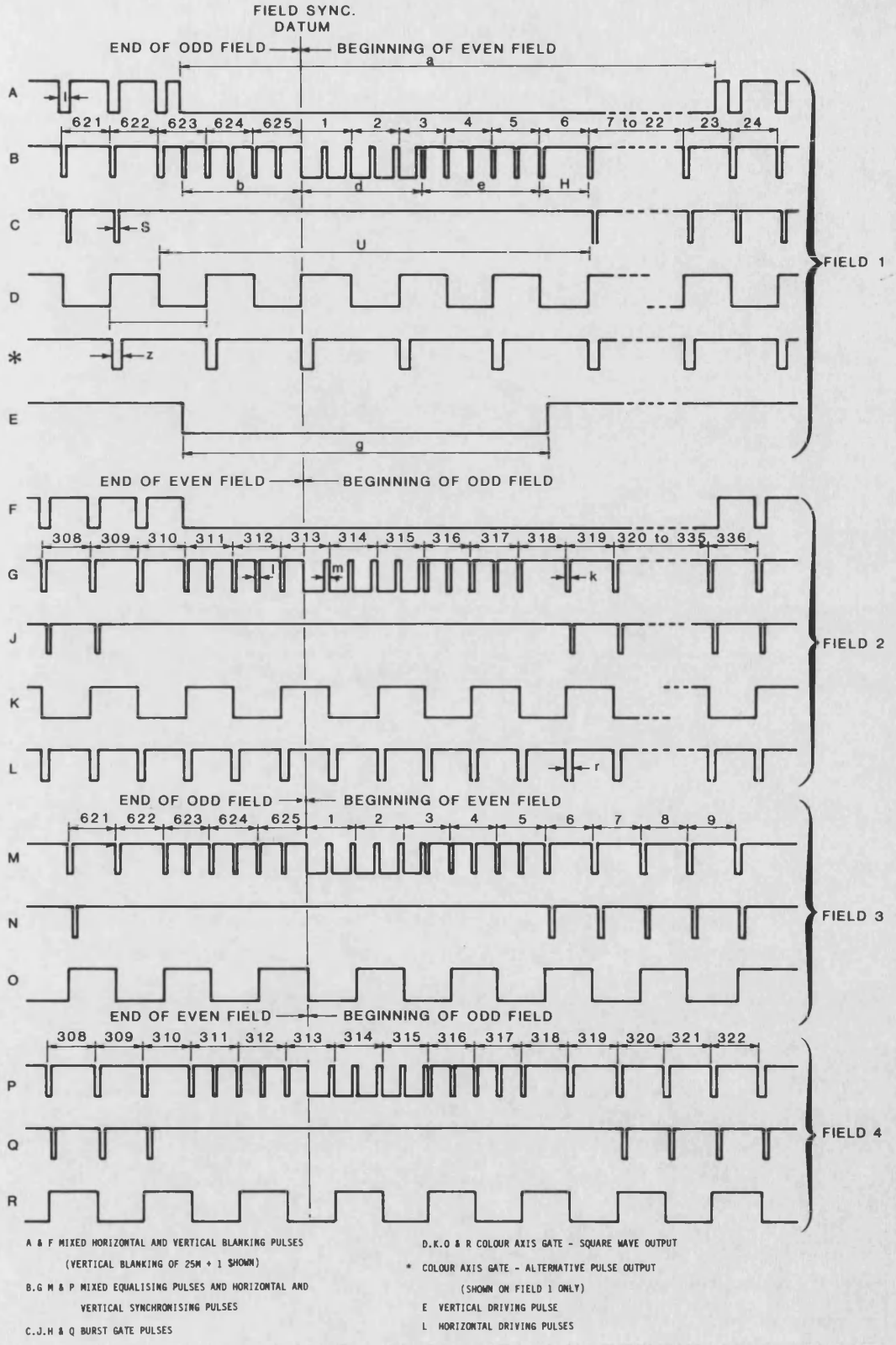


FIGURE 7 THE PAL 625 SYSTEM TIMING DIAGRAM

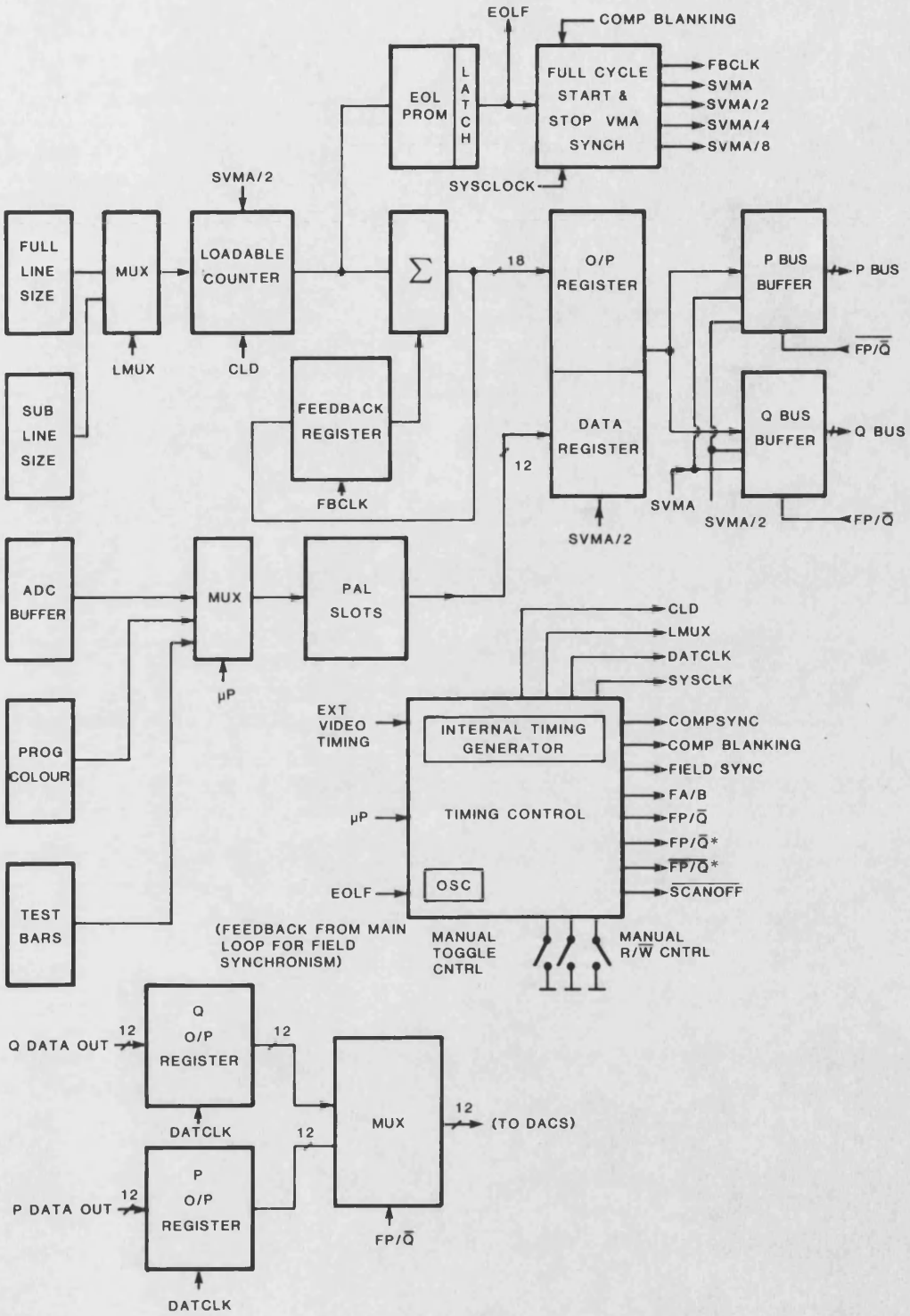
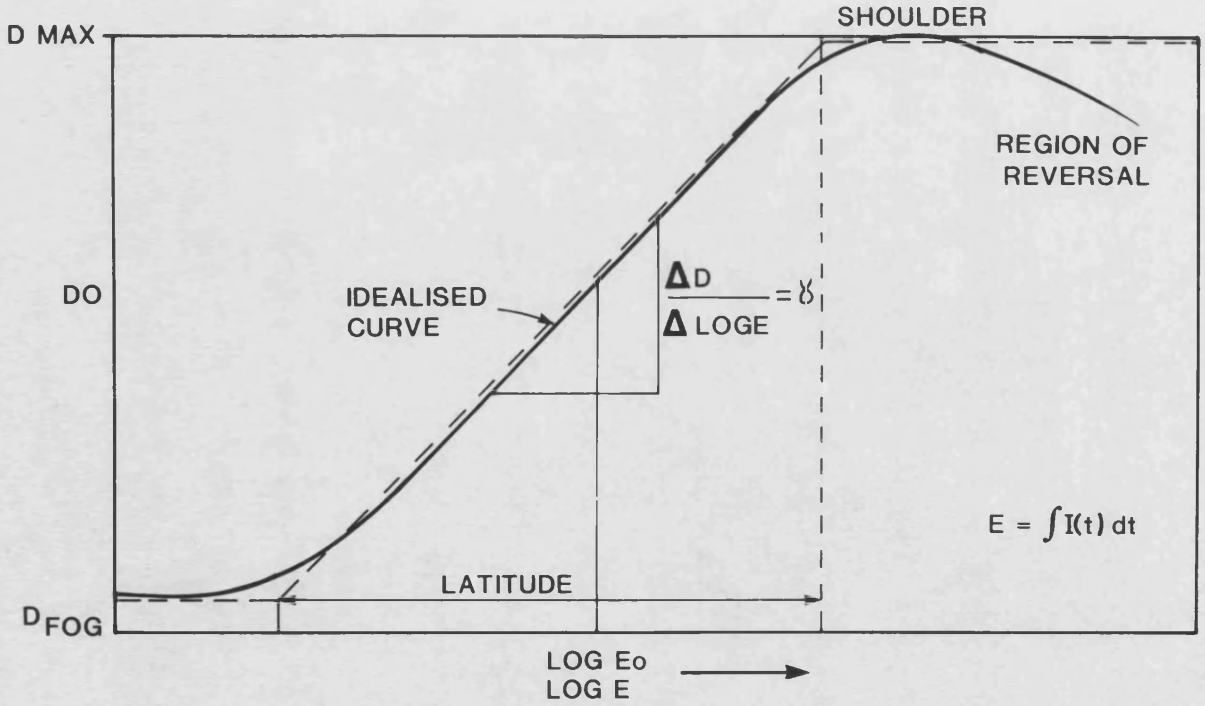


FIGURE 8 BLOCK DIAGRAM OF THE DIGITAL VIDEO SYSTEM



(HURTER & DRIFFIELD
EMULSION CHARACTERISTIC)

E = EXPOSURE

ϒ = REPRESENTS CONTRAST

I = LIGHT INTENSITY

LATITUDE = ABSCISSA LENGTH OF LINEAR PORTION

FIGURE 9 THE HURTER AND DRIFFIELD EMULSION CURVE

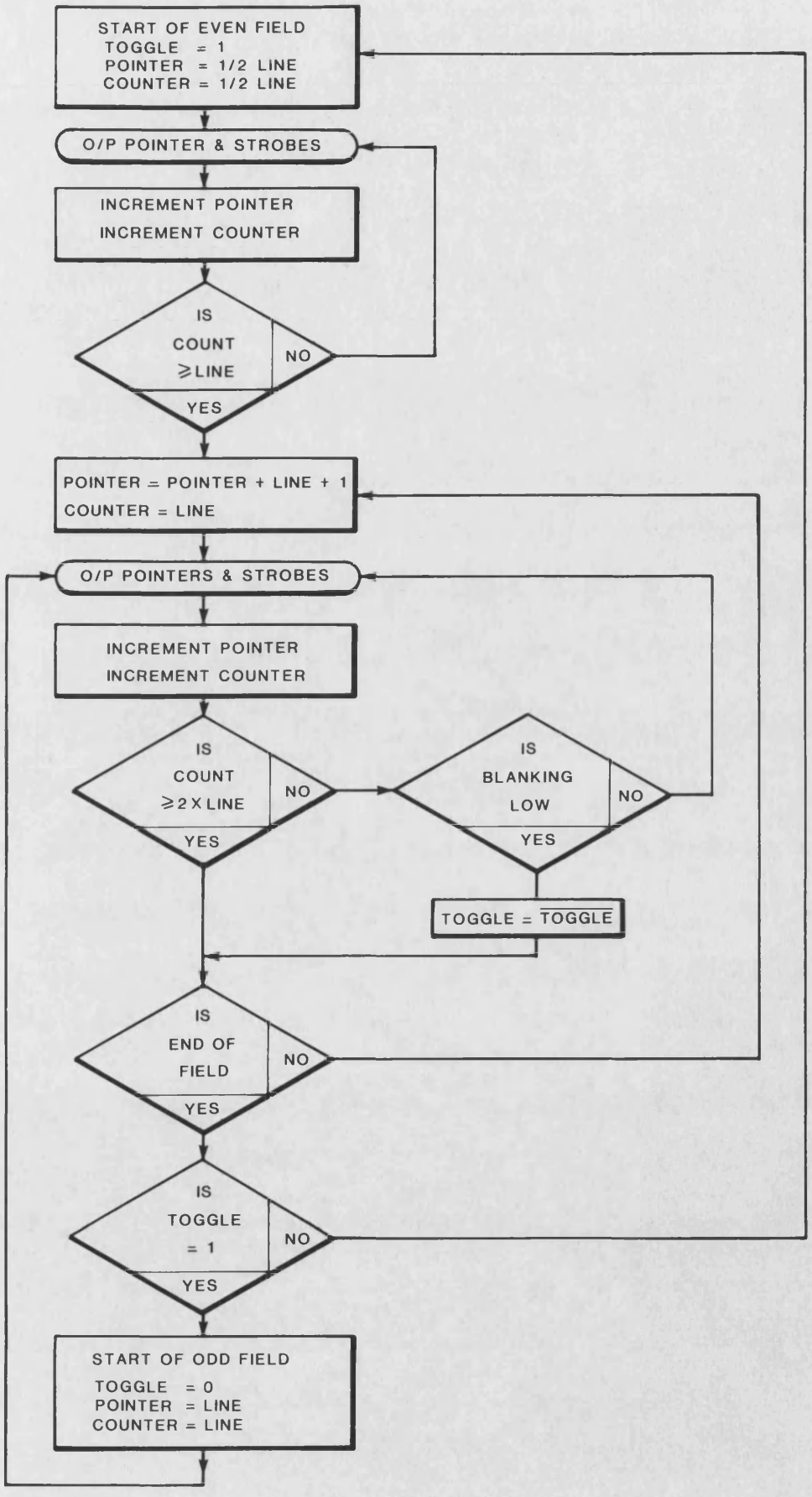
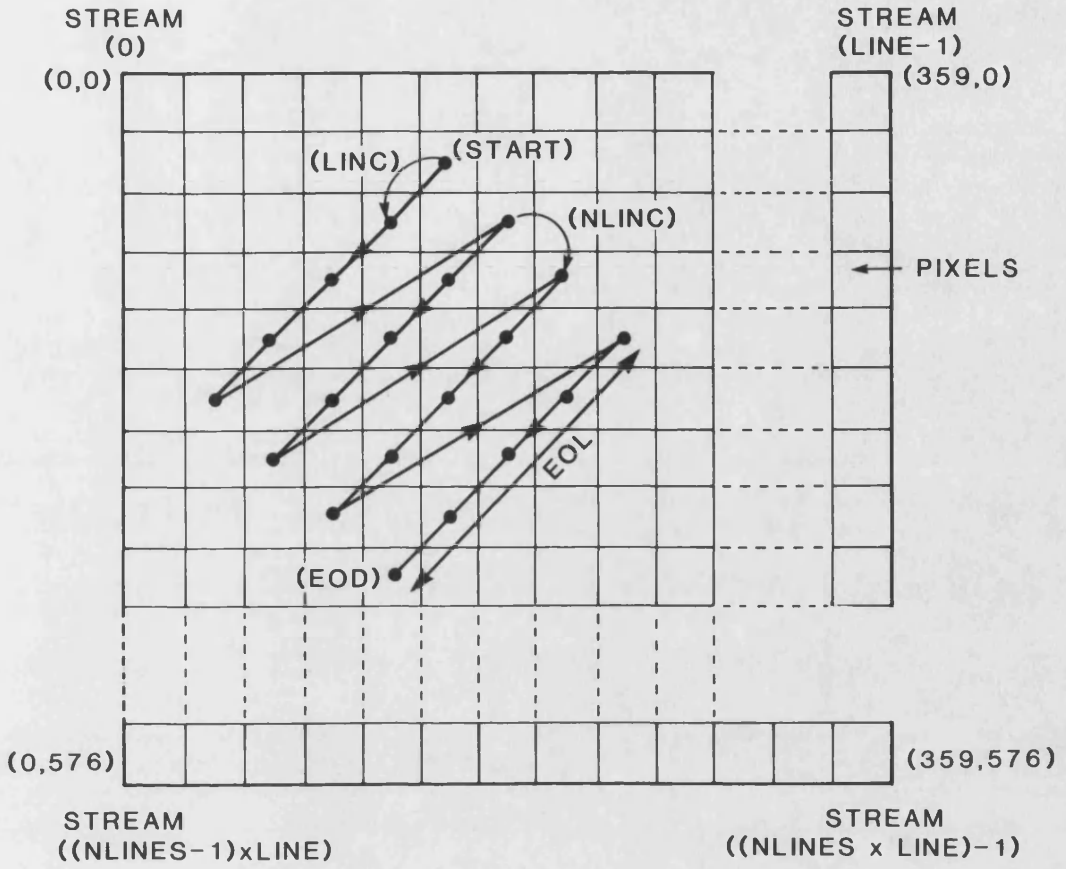


FIGURE 10 FLOWCHART FOR INTERLACED STORAGE OF DIGITAL VIDEO



SCAN PARAMETERS

*45° VECTORED SCAN BLOCK TRANSFER

*LINC = (LINE-1)

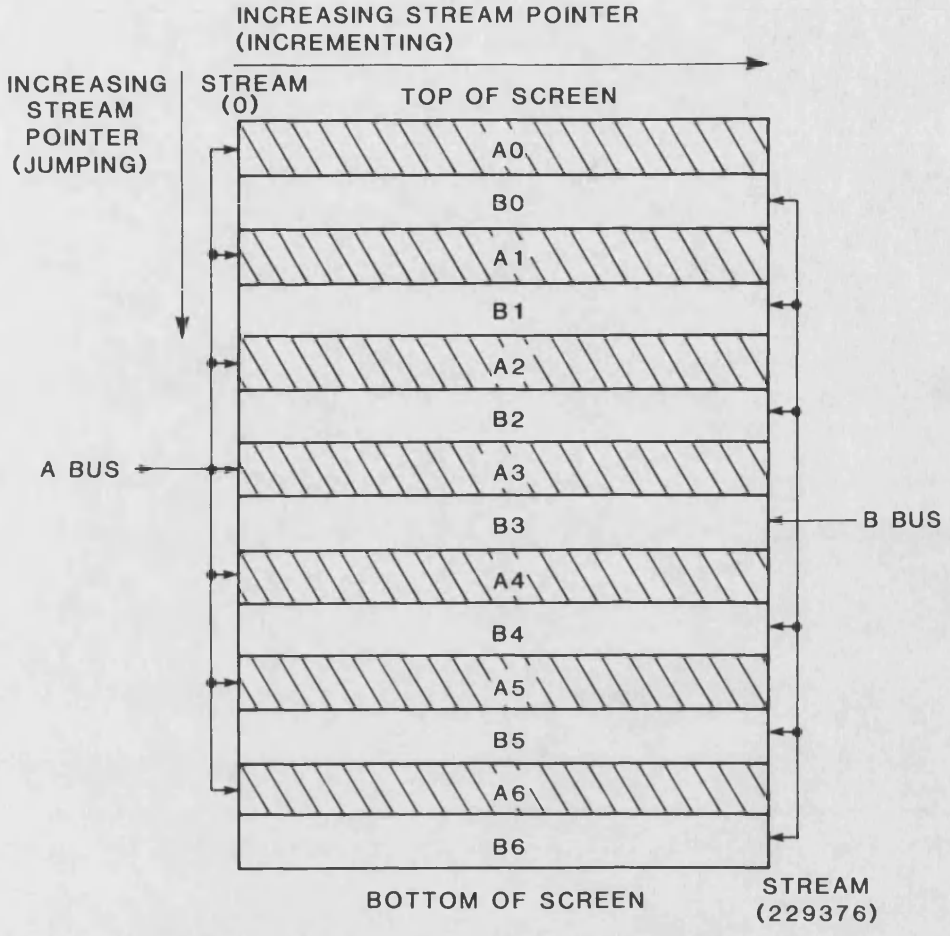
*NLINC = (LINE+1)

*EOL = 4

*EOD = 4(i.e. NO LINES)

*START = VARIABLE POSITION
(18 BIT ABSOLUTE ADDRESS)

FIGURE 11 MEMORY ADDRESSING USING THE VECTOR SCAN

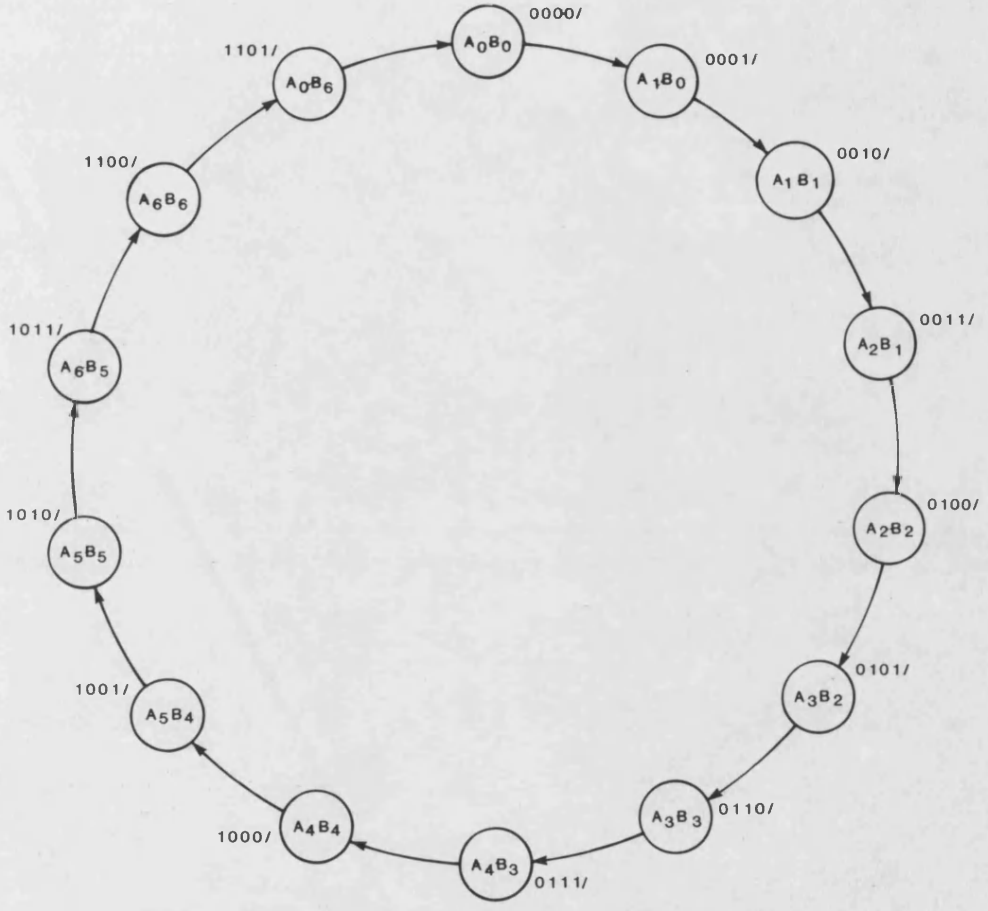


NB: EACH BAR REPRESENTS 1 RAM CHIP AND CONTAINS 16K LOCATIONS WHICH APPROXIMATES TO 45, 360 LENGTH, LINES.

FEATURES

* ALL RAMS HAVE COMMON ADDRESS(LSBS)
* BANKS A & B HAVE SEPARATE DATA BUSSES
* ALL CHIPS HAVE SEPARATE SELECT LINES FROM PREDICTORS.

FIGURE 12 INTERLACED MEMORY BARS



INPUTS: A17 A16 A15 A14/ (ADDRESS MSBS)
OUTPUTS: ENABLES TO RAM BANKS A0 →A6, B0 →B6
LETTERS IN CIRCLES INDICATE WHICH BANKS
ENABLED IN GIVEN STATE

FIGURE 13 POWER-UP PREDICTOR STATE DIAGRAM

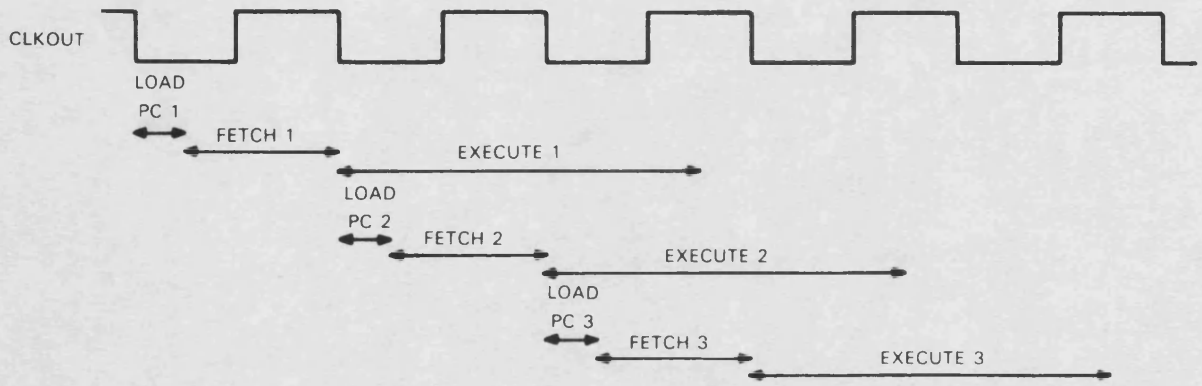


FIGURE 14 TMS32010 INSTRUCTION PREFETCH AND EXECUTION TIMING

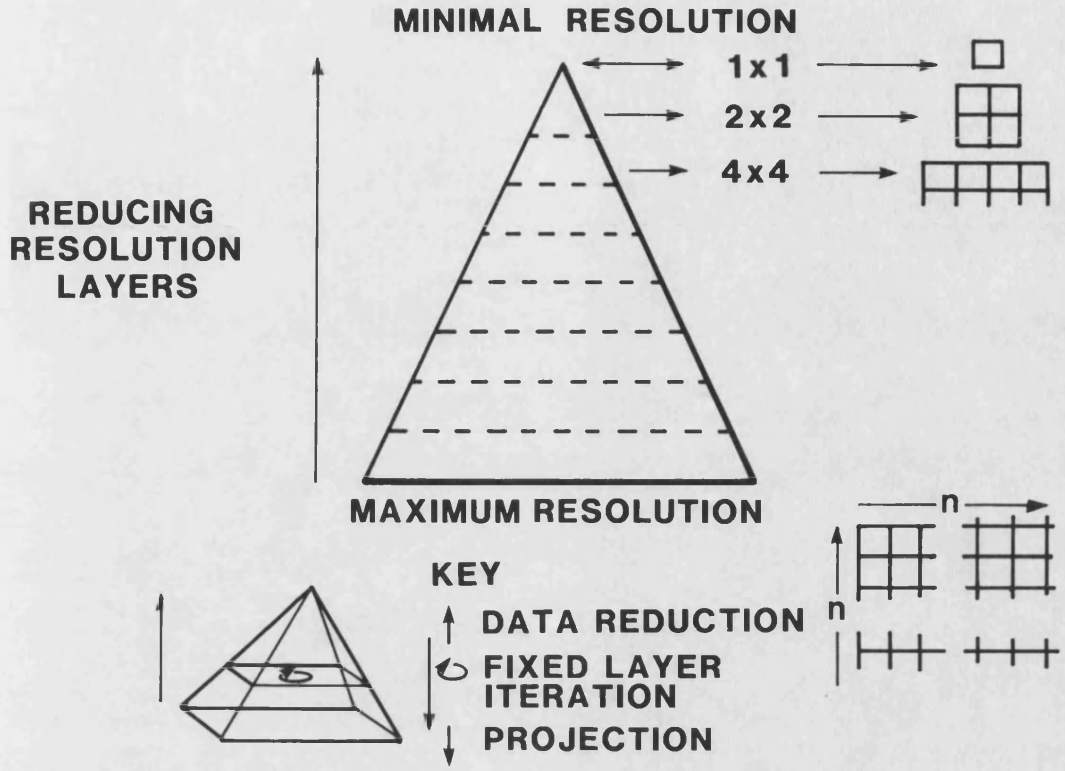


FIGURE 15 THE PYRAMID STRUCTURE

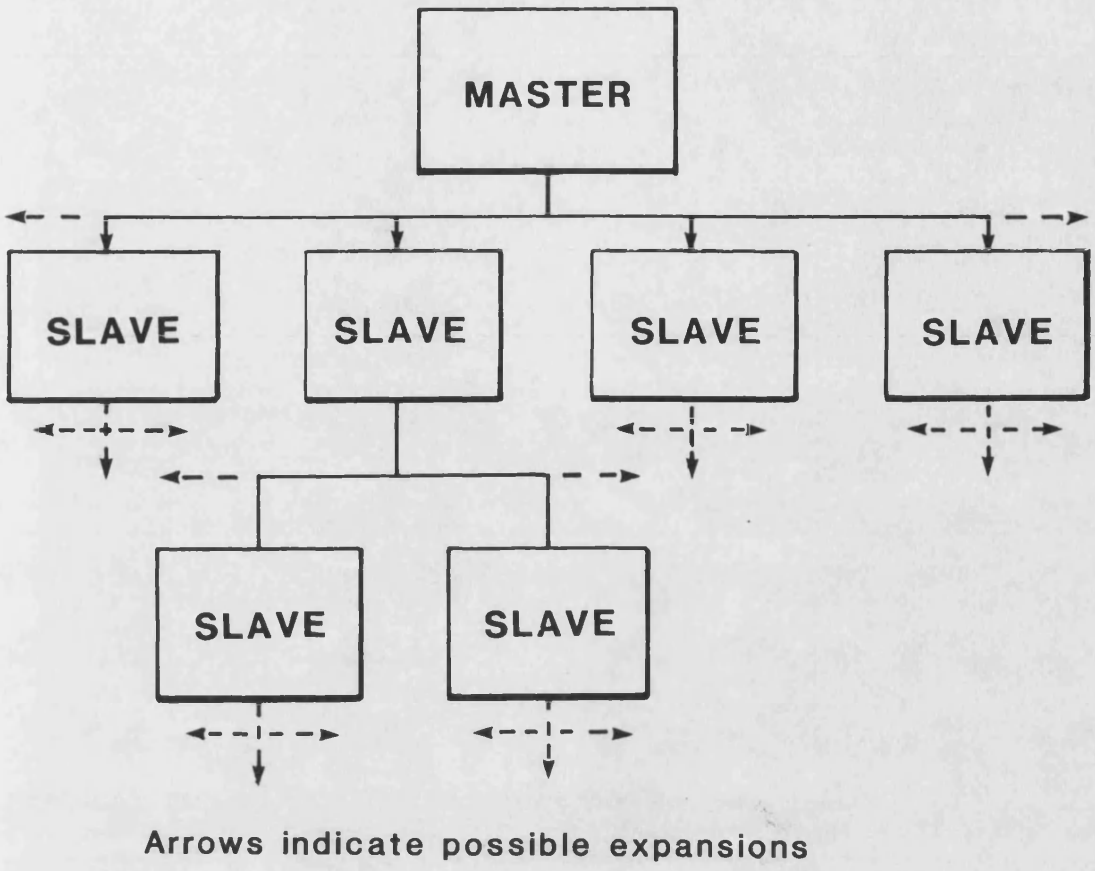


FIGURE 16 THE EXPANDING TREE STRUCTURE

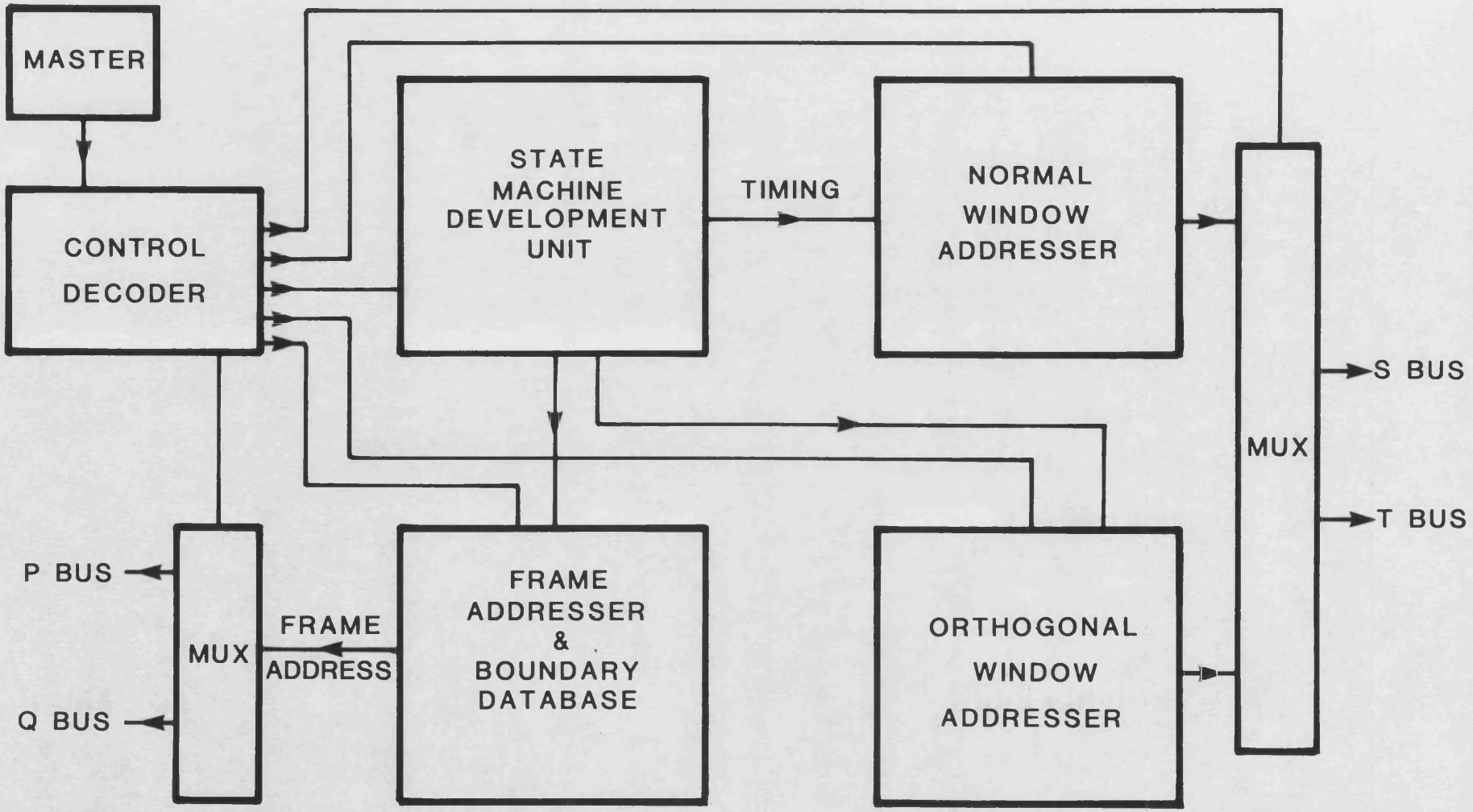
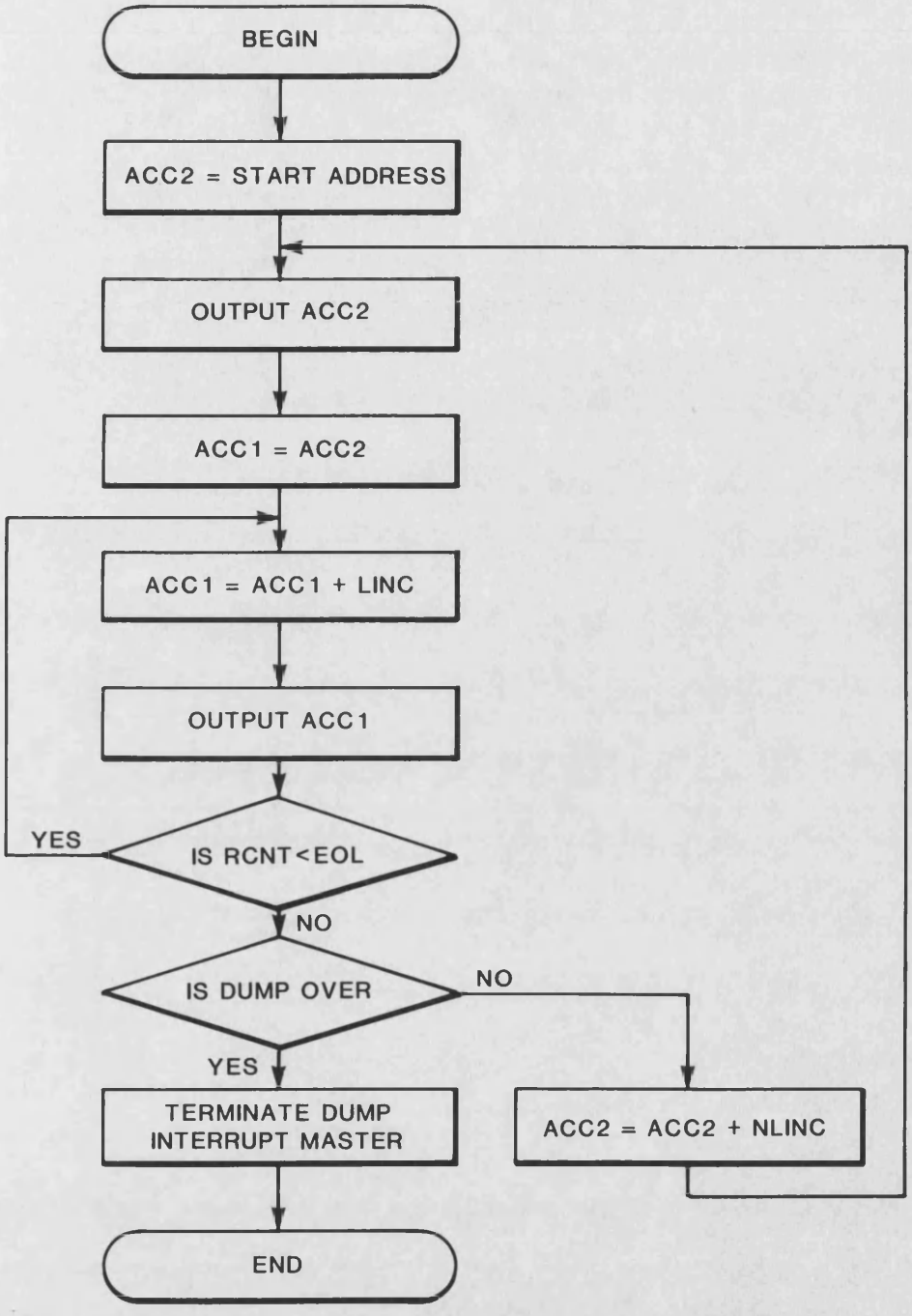


FIGURE 17 BLOCK DIAGRAM OF THE VECTOR ORTHOGONAL SCANNER (VOS)



N.B.

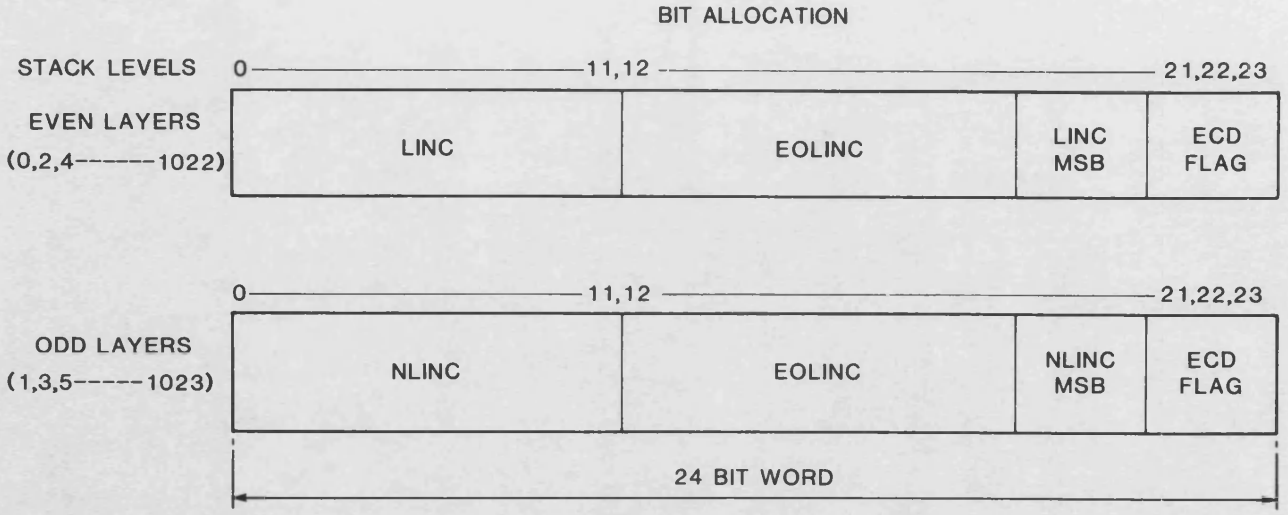
START = ABSOLUTE START ADDRESS, 18 BIT(0-262144)

LINC = LINE INCREMENT, 13 BIT(0-8191)

NLINC = NEW LINE INCREMENT, 13 BIT(0-8191)

RCNT = RUNNING COUNT, 10 BIT(0-1023)

FIGURE 18 FLOWCHART FOR FRAME STORE ADDRESS GENERATION USING THE VOS



NOTE:

LINC = LINE INCREMENT

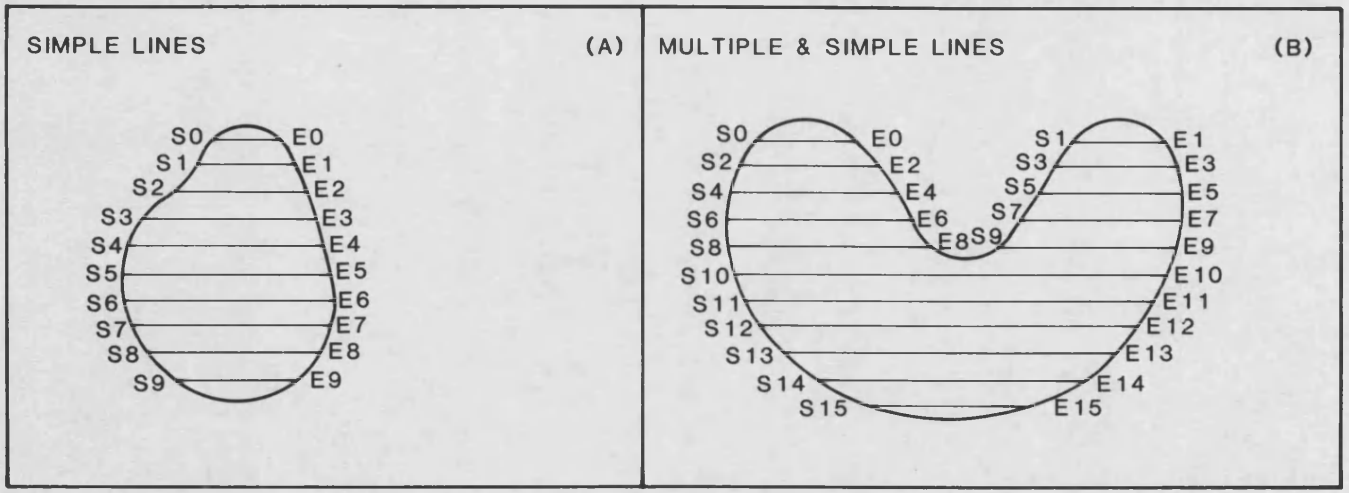
NLINC = NEW LINE INCREMENT

EOLINC = LINE COUNT IN PIXELS (SAME IN ODD & EVEN LAYERS)

LINCMSB = }
NLINCMSB = } AUXILLARY BIT USED TO EXTEND JUMP RANGES TO 8K

EOD FLAG = END OF DUMP FLAG USED TO TERMINATE TRANSFERS (SAME IN ODD & EVEN LAYERS)

FIGURE 19 DATA CONFIGURATION OF VOS RAM BANK



Sm = START OF LINE m
Em = END OF LINE m

A REQUIRES (10x2) = 20 STACK LEVELS

B REQUIRES (10x2) + (6x2) = 32 STACK LEVELS

FIGURE 20 SIMPLE AND MULTIPLE SCAN LINES

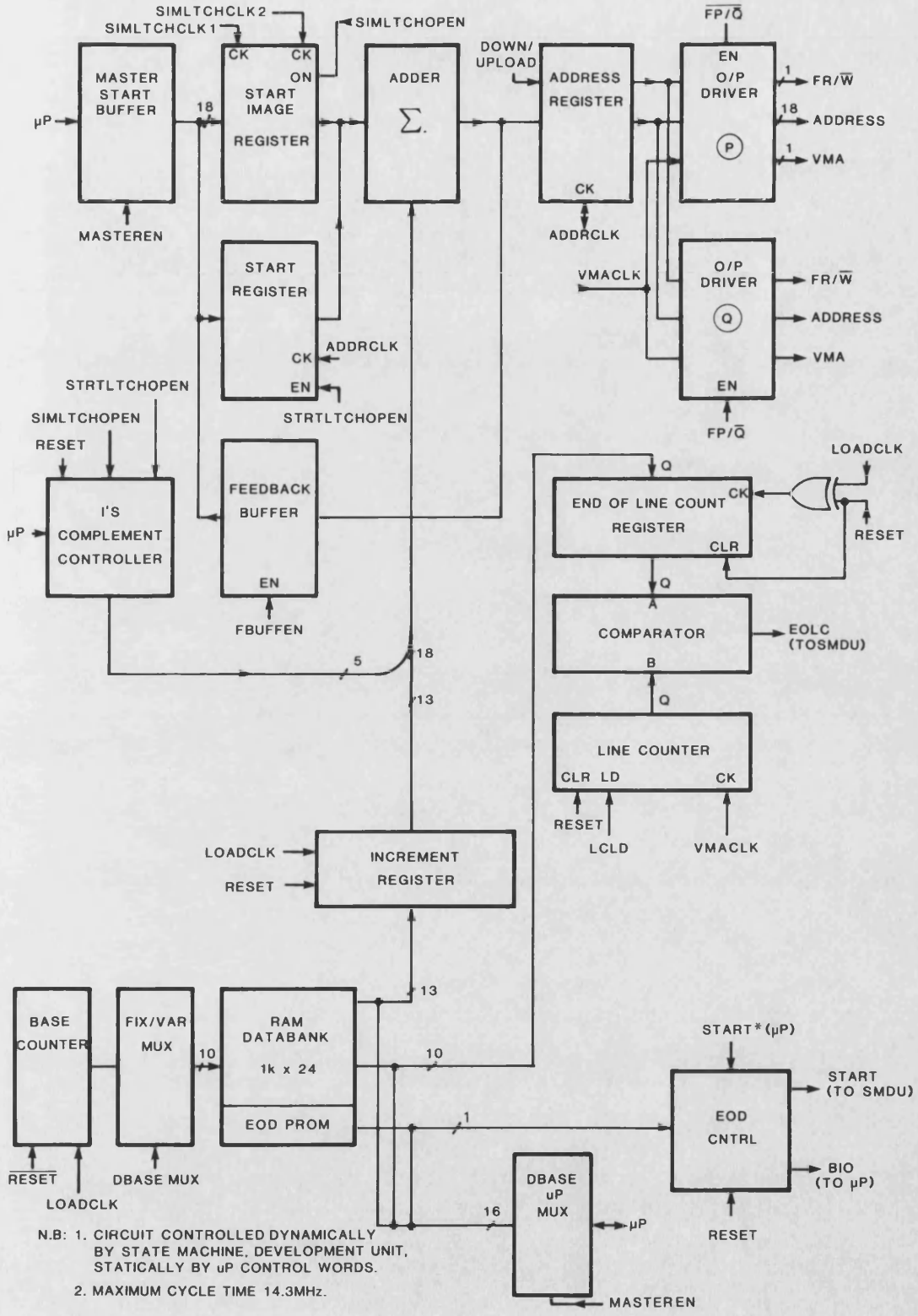
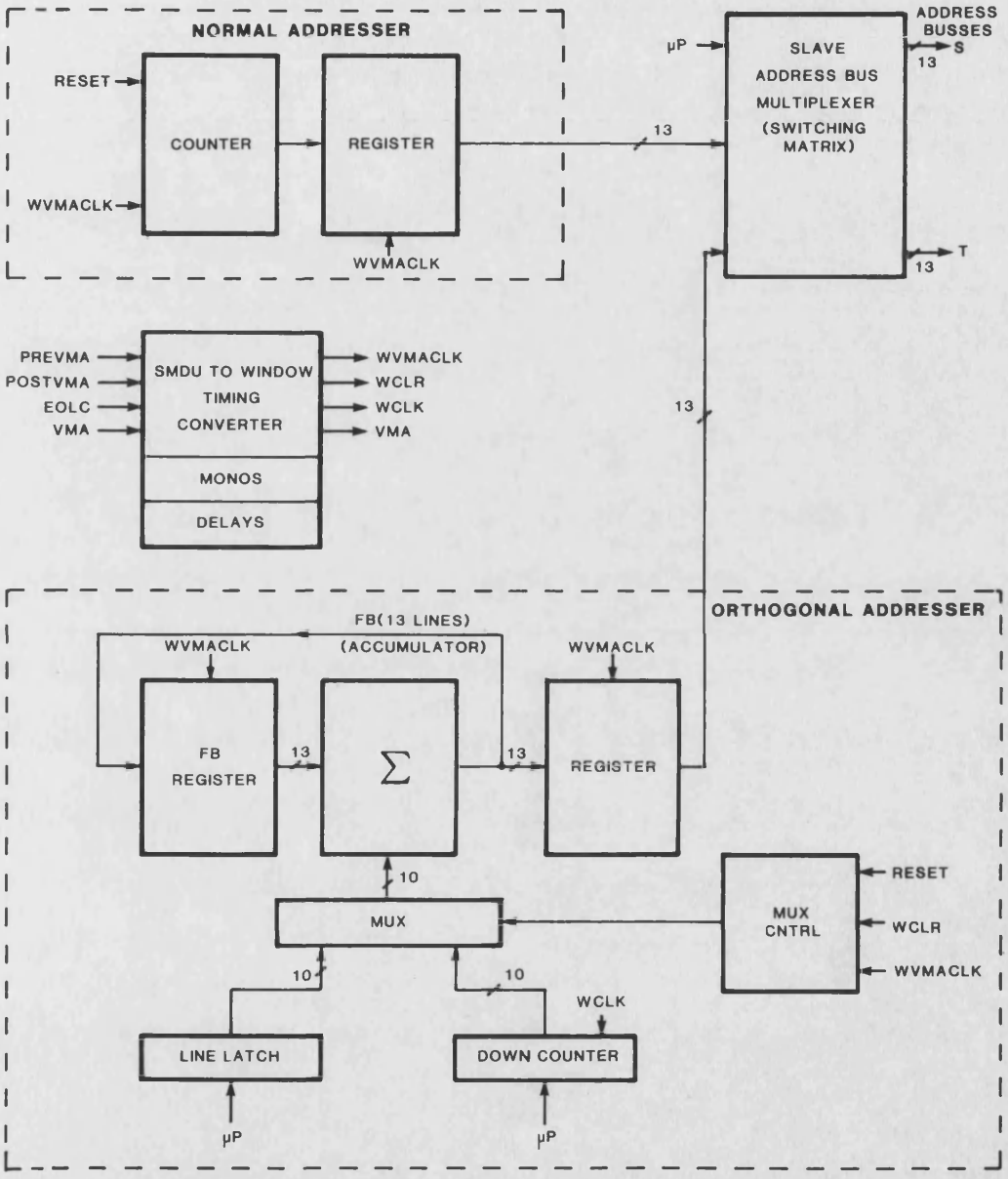
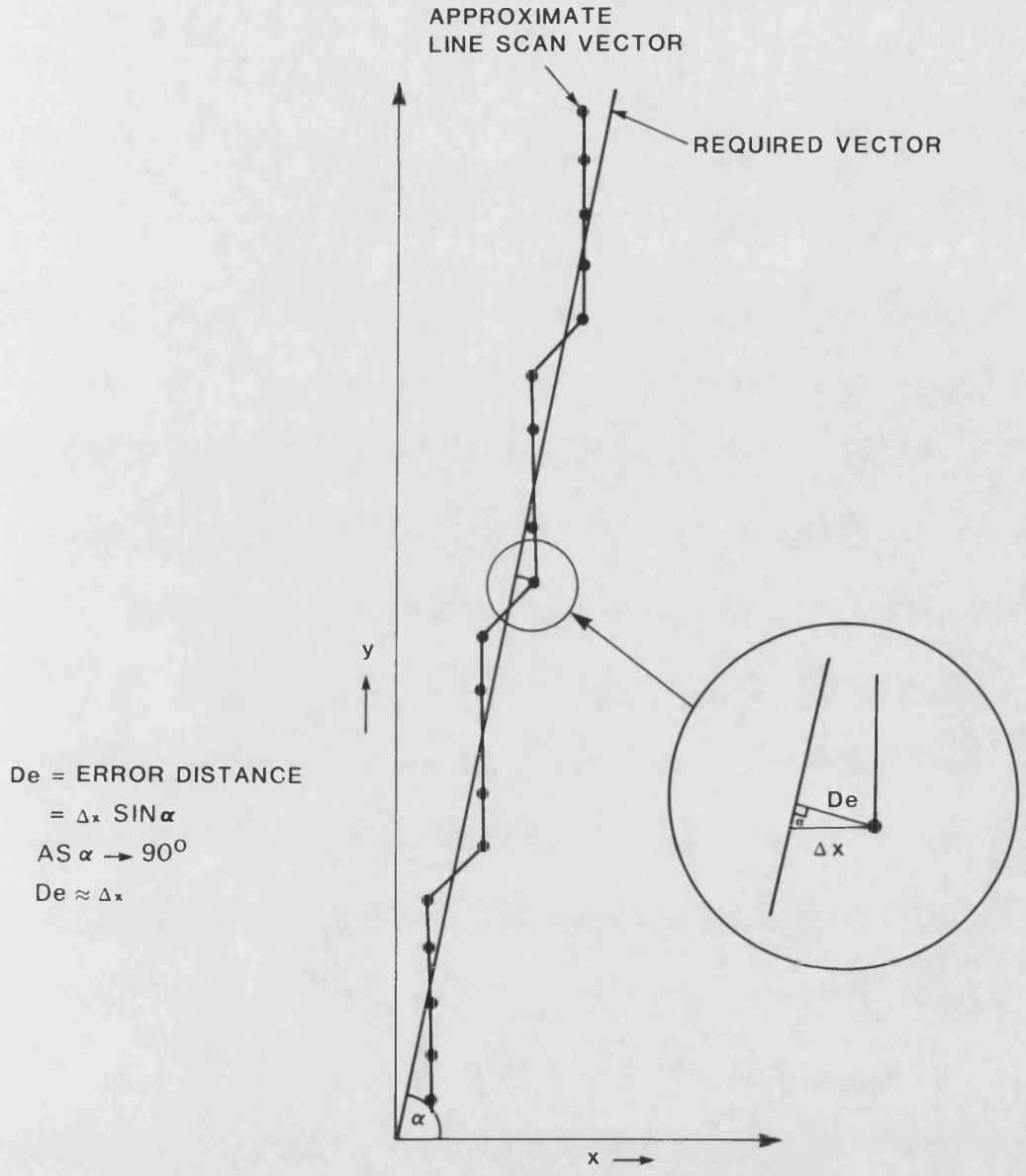


FIGURE 21 BLOCK DIAGRAM OF THE FRAME ADDRESSER



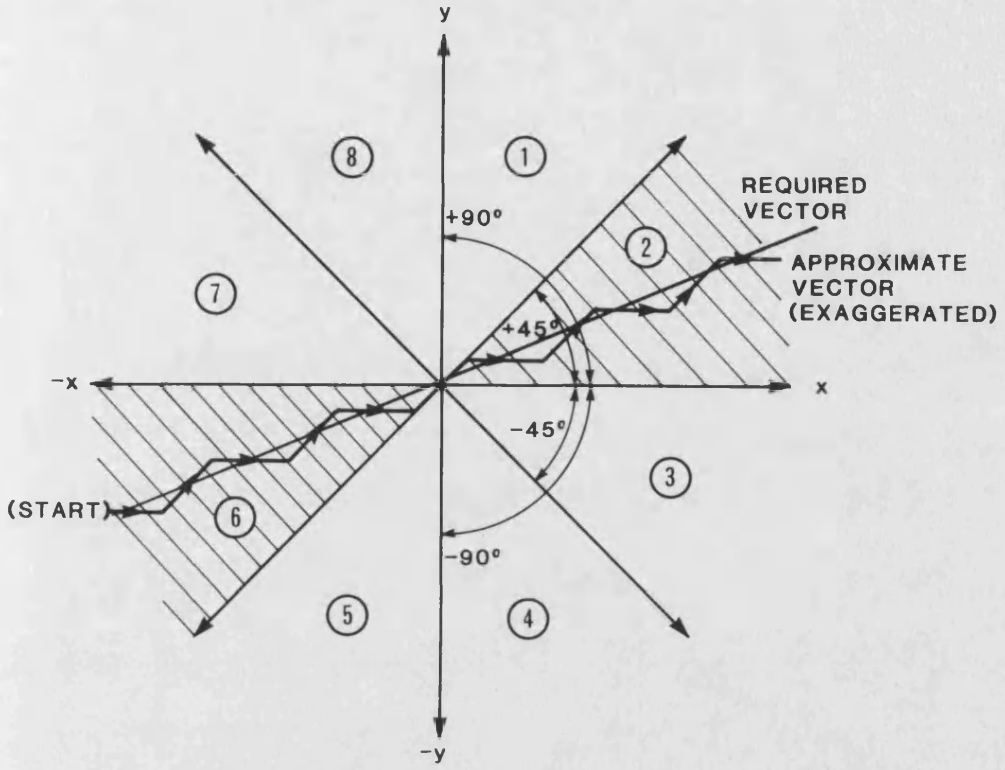
N.B: μP INDICATES CONTROLLED (NORMALLY LATCHED) FROM MASTER PROCESSOR

FIGURE 22 BLOCK DIAGRAM OF THE WINDOW ADDRESSER



NB: (x) to (y) RATIO OF STEP WISE, SCAN & START OFFSET
CHOSEN TO KEEP De SMALL (≤ 1.5 units)

FIGURE 23 LINE MODE VECTOR SCANNING



NB: VECTORS THROUGH SECTORS 6&2 BEGIN AT THE BOTTOM-MOST THEN LEFT-MOST POSITION.

ALL OTHER SECTORS USE TOP-MOST THEN LEFT-MOST START POINT.

FIGURE 24 LINE SCAN VECTOR SPACE

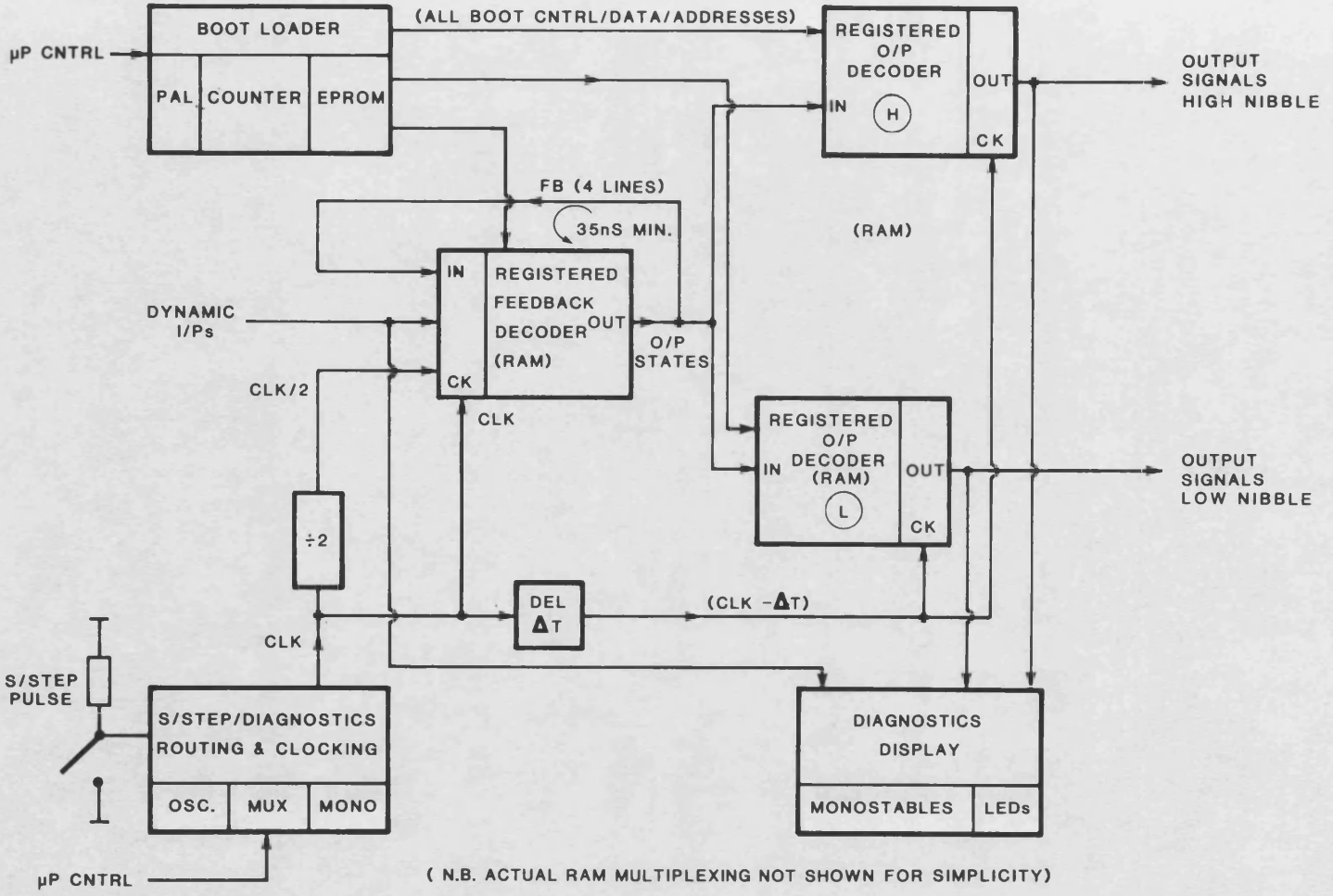


FIGURE 25 THE STATE MACHINE DEVELOPMENT UNIT (SMDU)

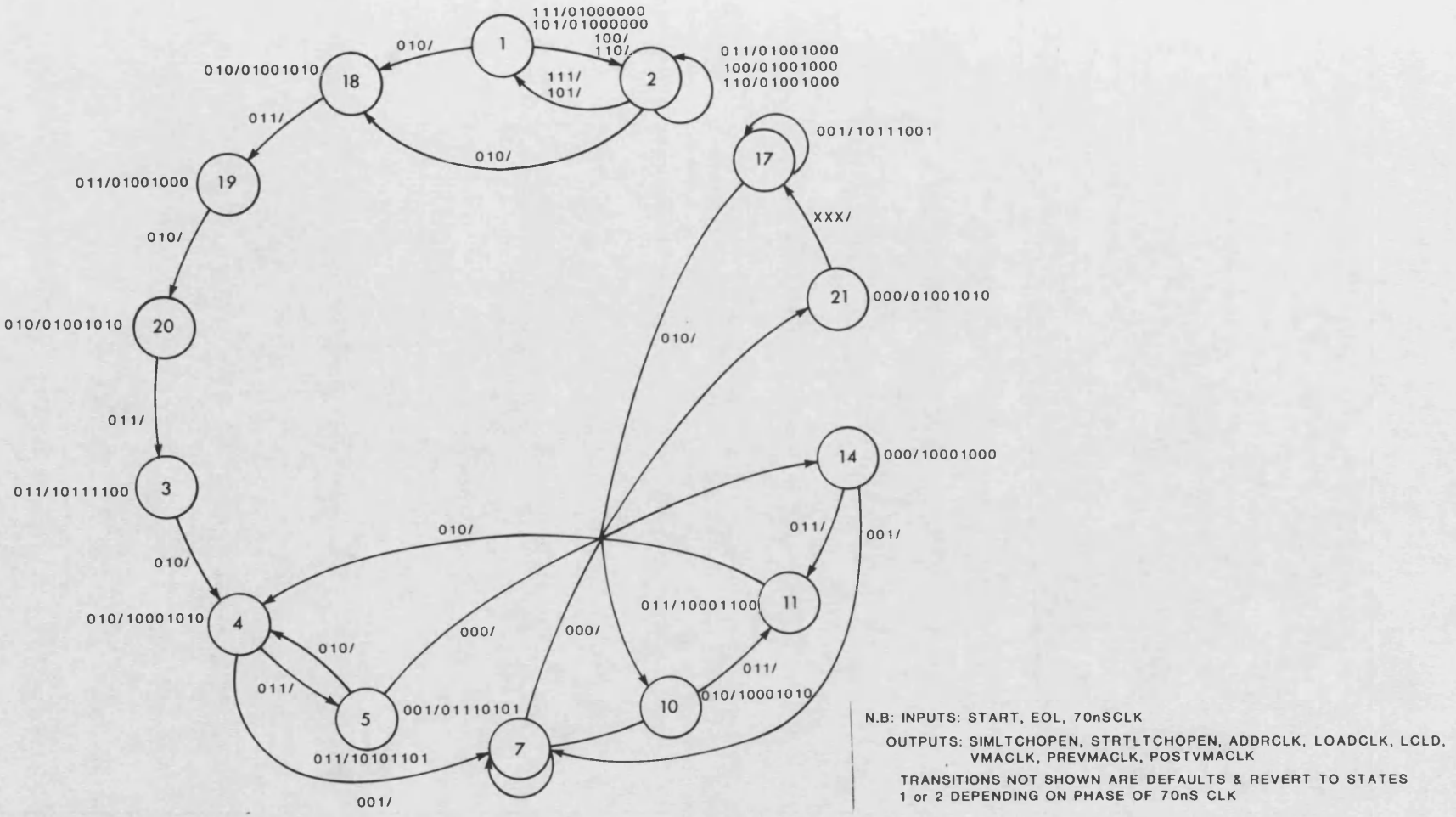


FIGURE 26 STATE DIAGRAM FOR THE SMDU

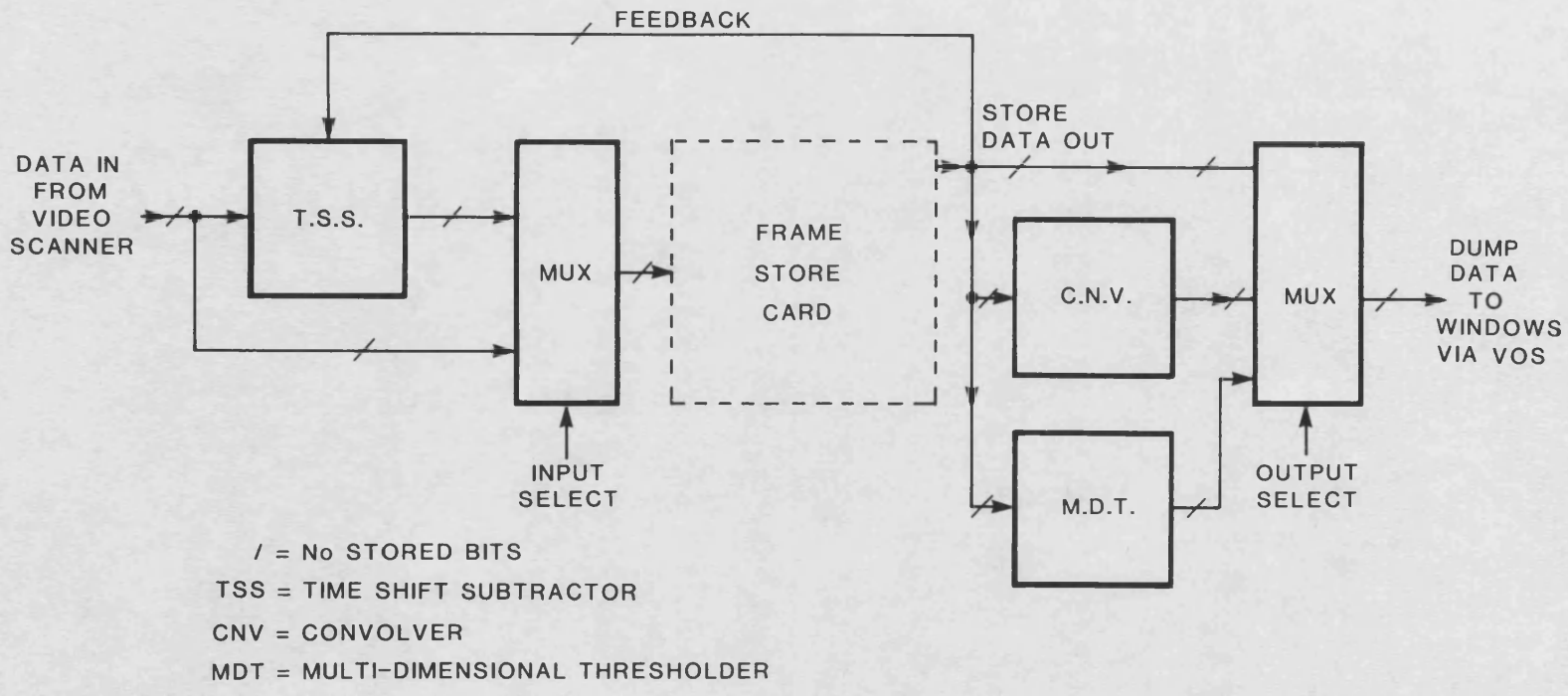
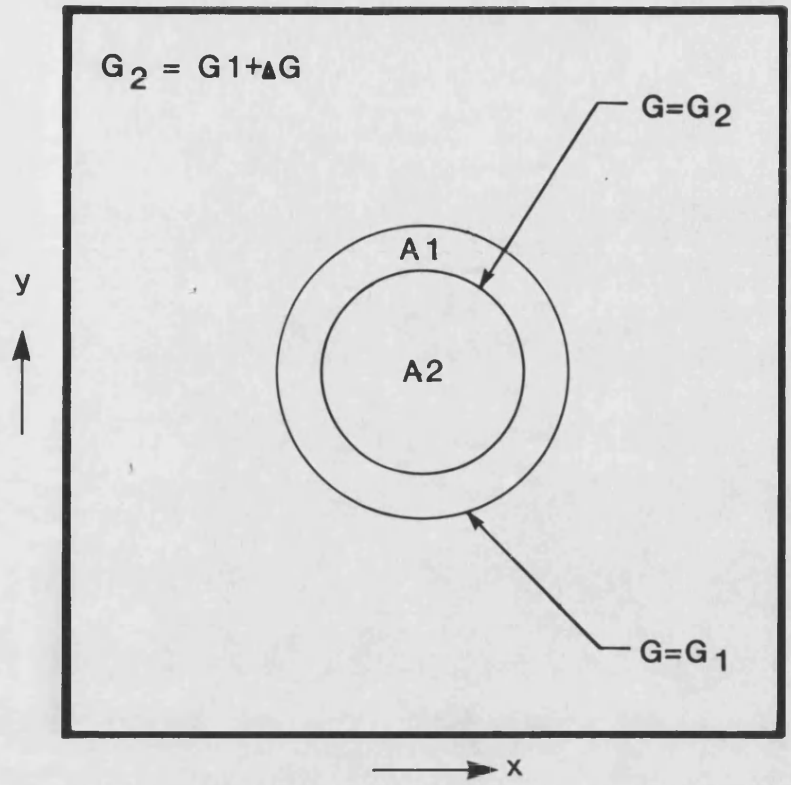
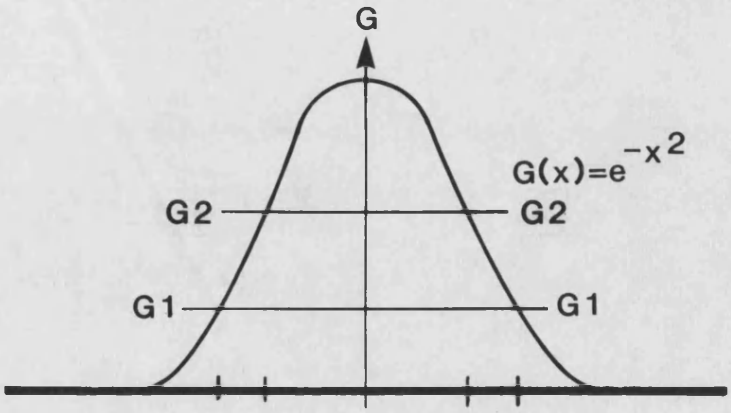


FIGURE 27 THE DATA TRANSFER PROCESSOR (DTP)

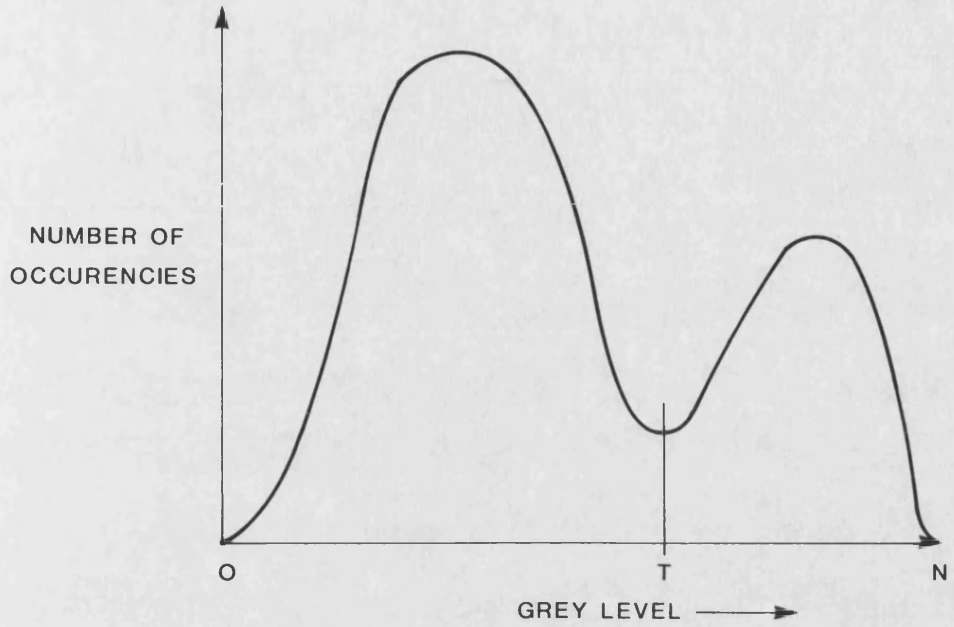


a) 2-D GAUSSIAN, CONTOUR LINES IN IMAGE



b) SIDEWAYS VIEW OF 2-D GAUSSIAN

FIGURE 28



T = OPTIMUM THRESHOLD LEVEL
GREY LEVELS RANGE FROM O TO N

NB: IF HISTOGRAM NORMALISED WITH RESPECT TO PICTURE AREA THEN EQUIVALENT TO PROBABILITY DENSITY FUNCTION P.D.F.

FIGURE 29 THE BIMODAL HISTOGRAM

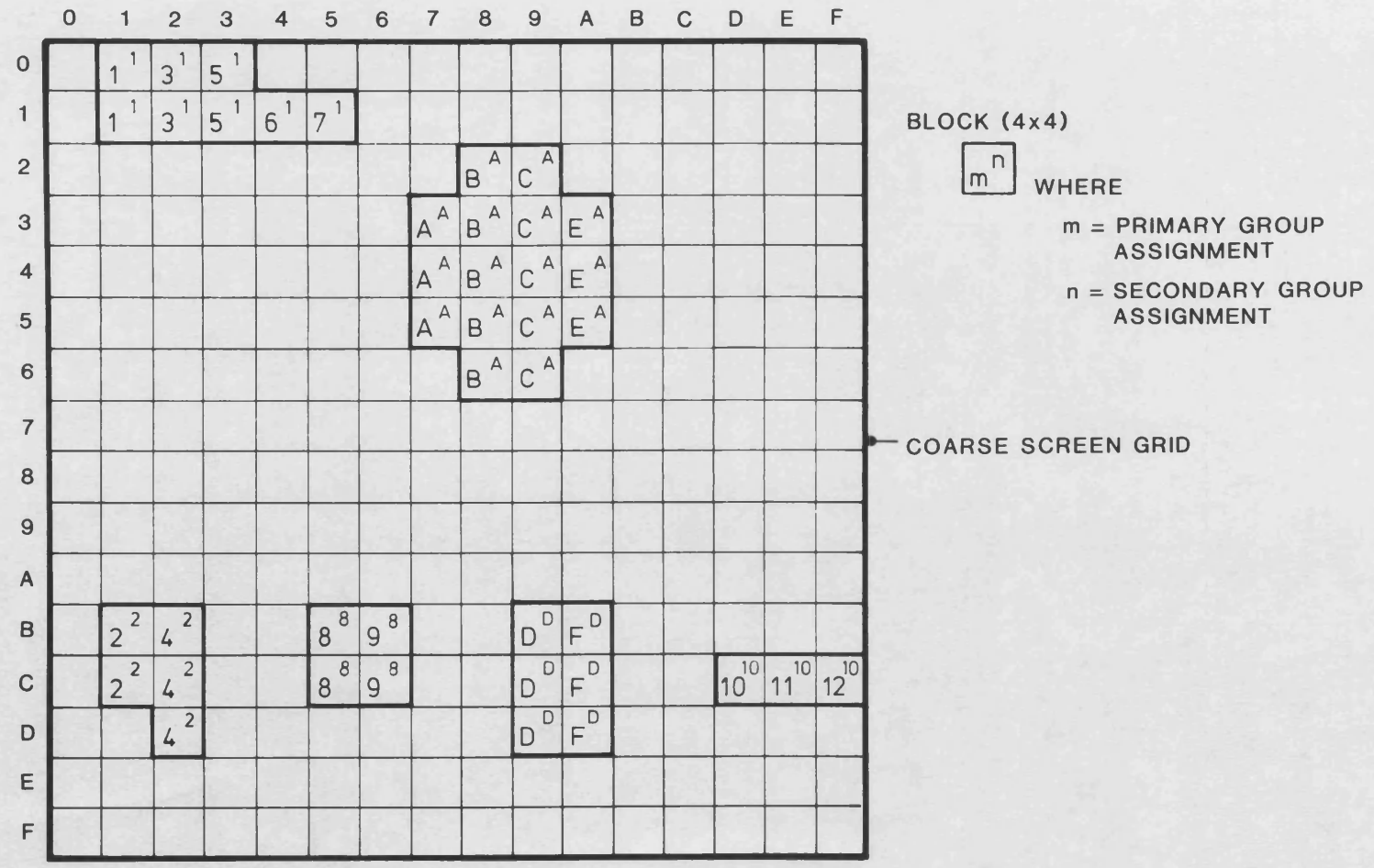


FIGURE 30 BLOB SEGMENTATION

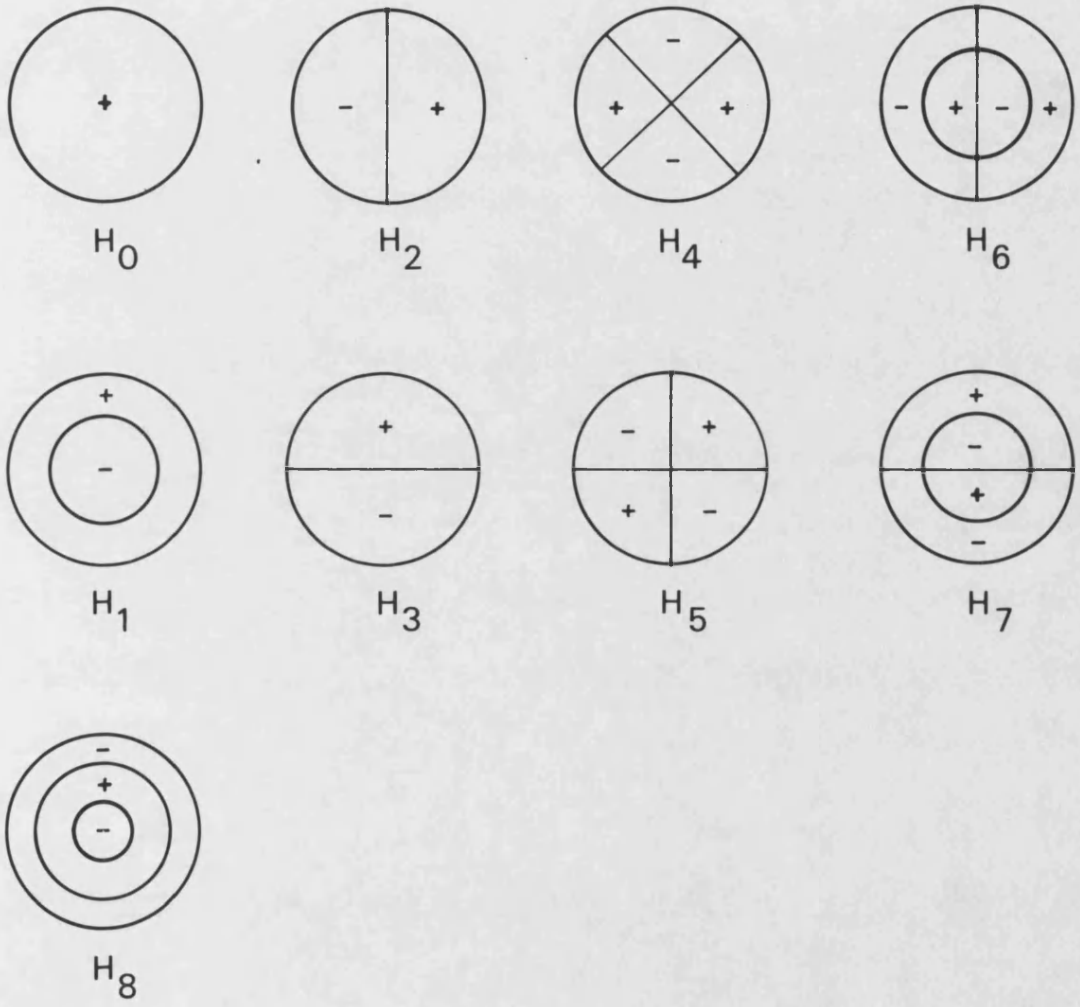
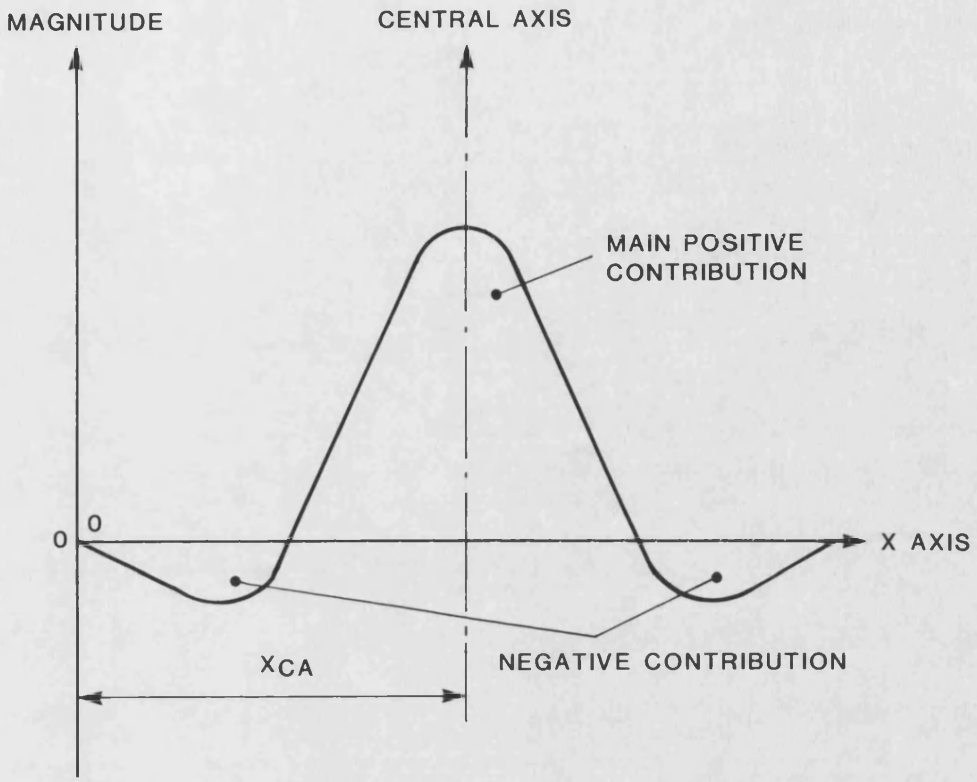
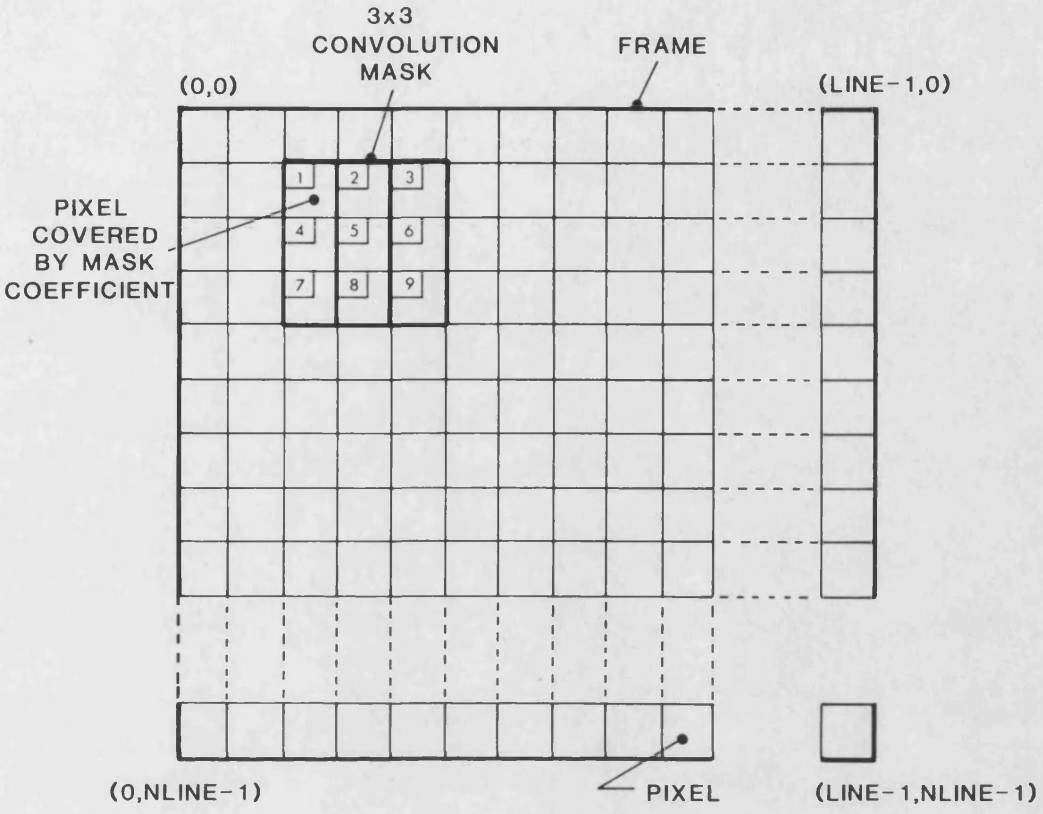


FIGURE 31 HEUCKEL EDGE MODEL SET



NB: CENTRAL AXIS SHOWN OFFSET FROM ORIGIN BY X_{CA}

FIGURE 32 LAPLACIAN FUNCTION



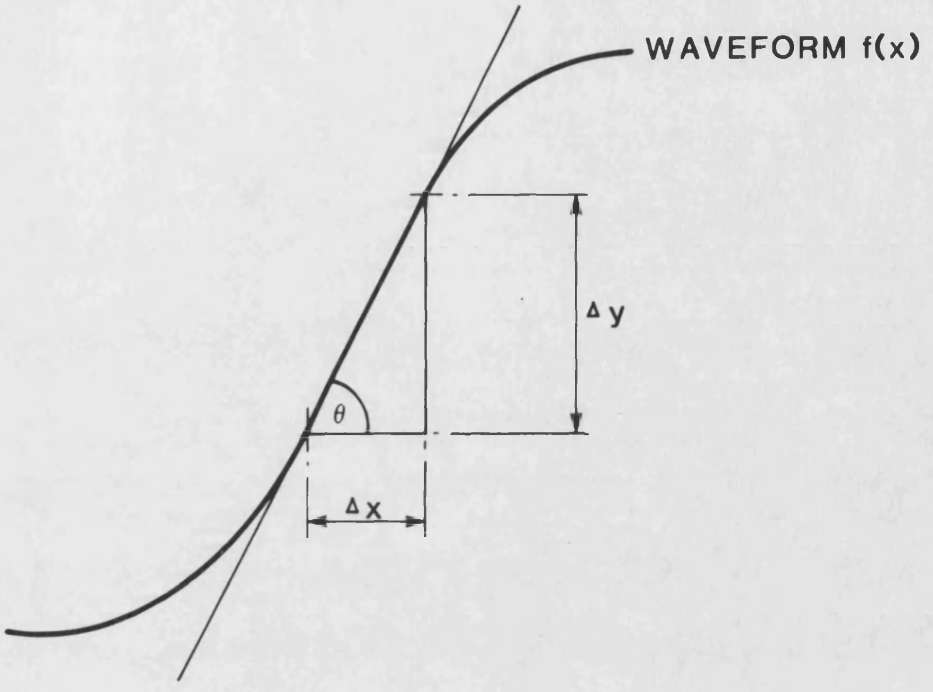
Vm = COEFFICIENT m VALUE (m = 1 to 9)

Pm = PIXEL m VALUE (m = 1 to 9)

$$O/P = \sum_{m=1}^{9} Vm.Pm \quad \text{FOR EACH MASK POSITION}$$

MASK IS STEPPED RIGHTWARDS UNTIL END OF LINE THEN FLYBACK , STEPS DOWN ONE LINE & REPEATS.

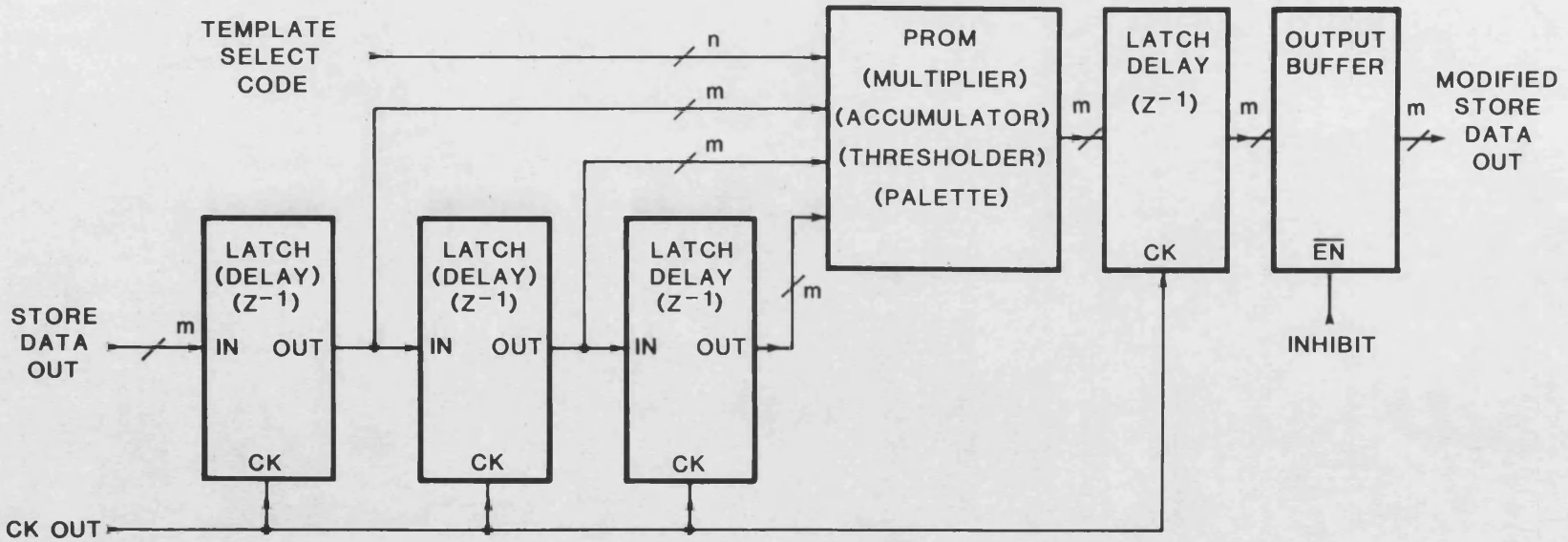
FIGURE 33 2-D CONVOLUTION FILTER EXAMPLE



$$MAG(x) = (\Delta y^{2} + \Delta x^{2})^{1/2}$$

$$ANGLE(x) = TAN^{-1}\left(\frac{\Delta y}{\Delta x}\right)$$

FIGURE 34 THE SIMPLE GRADIENT



m = NUMBER OF BITS FOR STORE DATA

n = NUMBER OF SEL CODE BITS

FIGURE 35 THE DTP TEMPLATE CONVOLVER

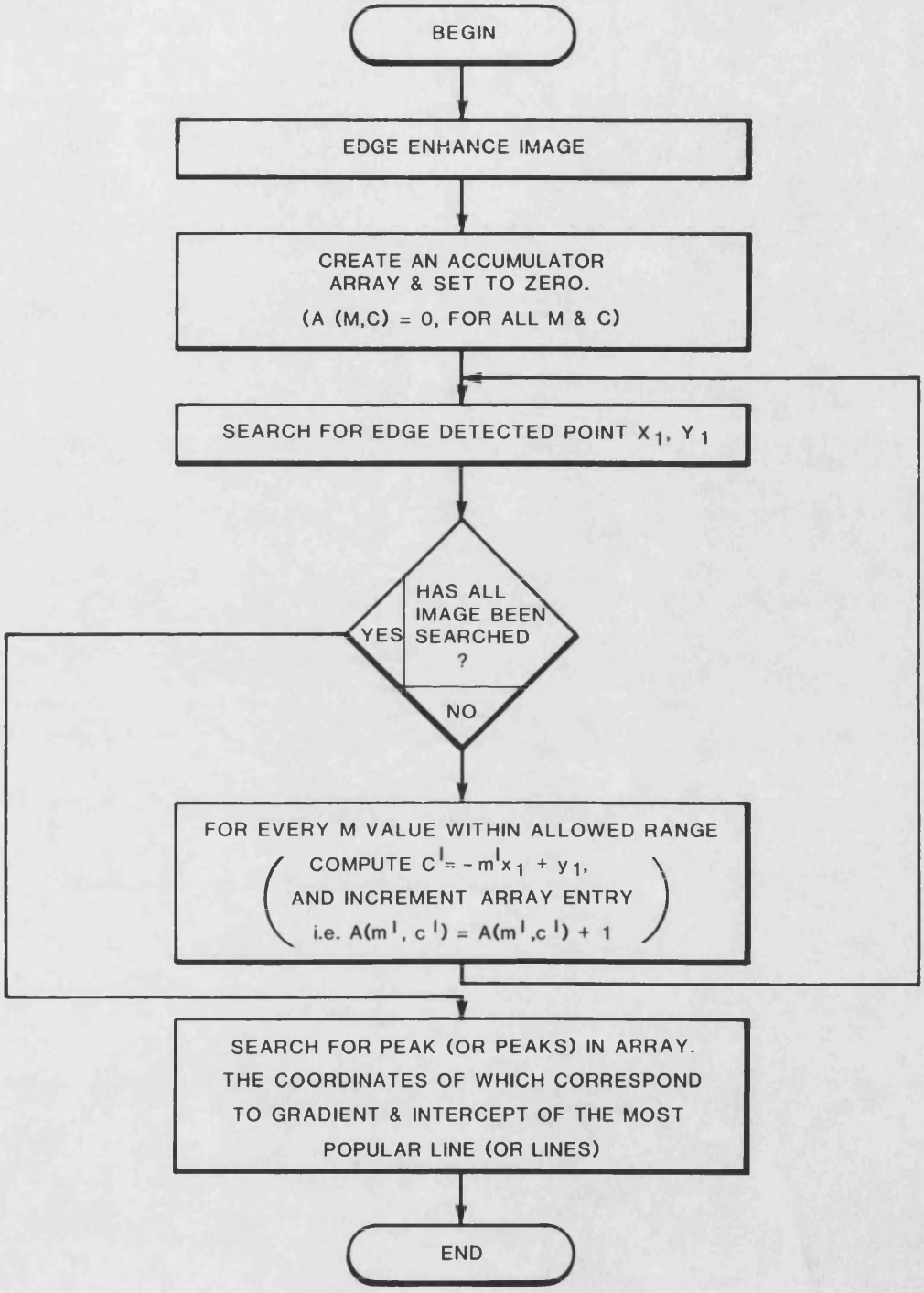
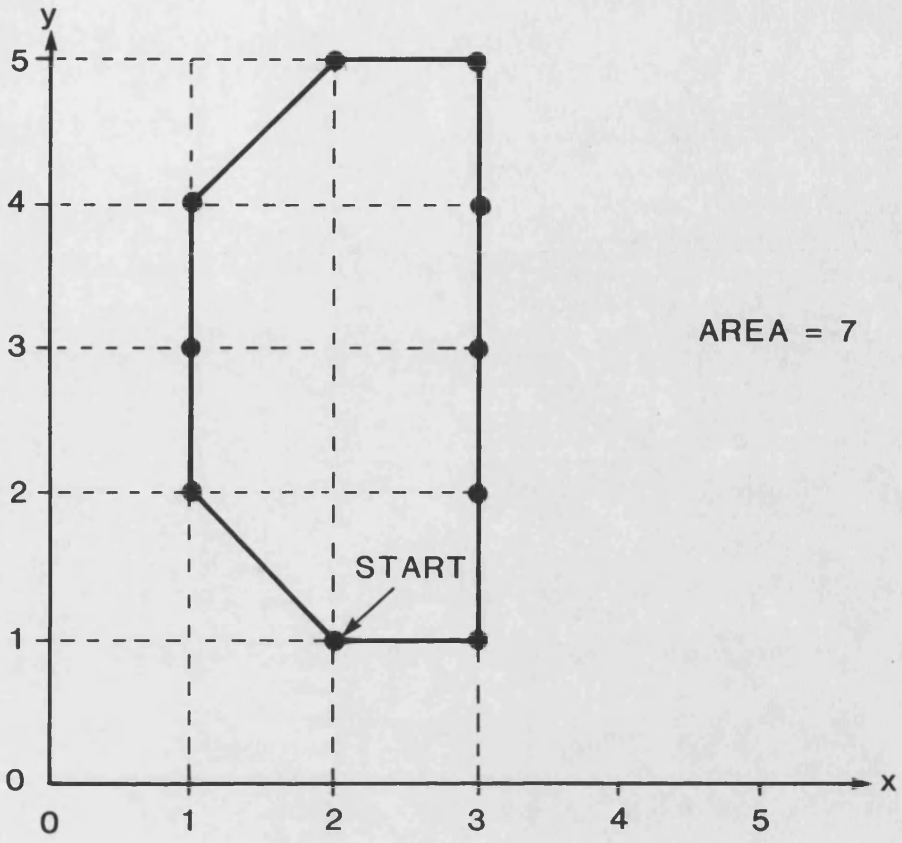


FIGURE 36

FLOWCHART FOR FINDING THE EQUATIONS OF ENHANCED LINES, USING THE HOUGH TRANSFORM

a) CHAIN CODING EXAMPLE



(CHAIN CODE ≡ 3221066664)

b) CHAIN CODING MASK

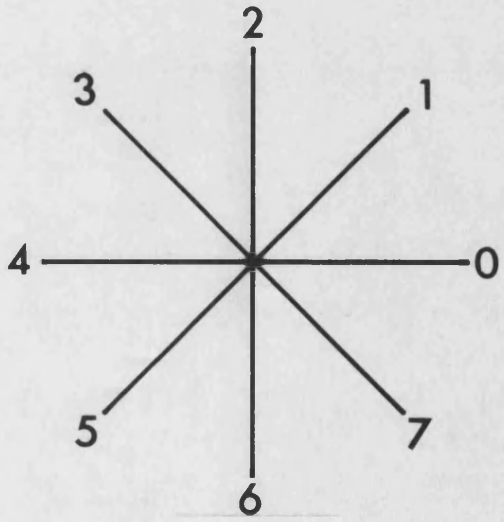


FIGURE 37

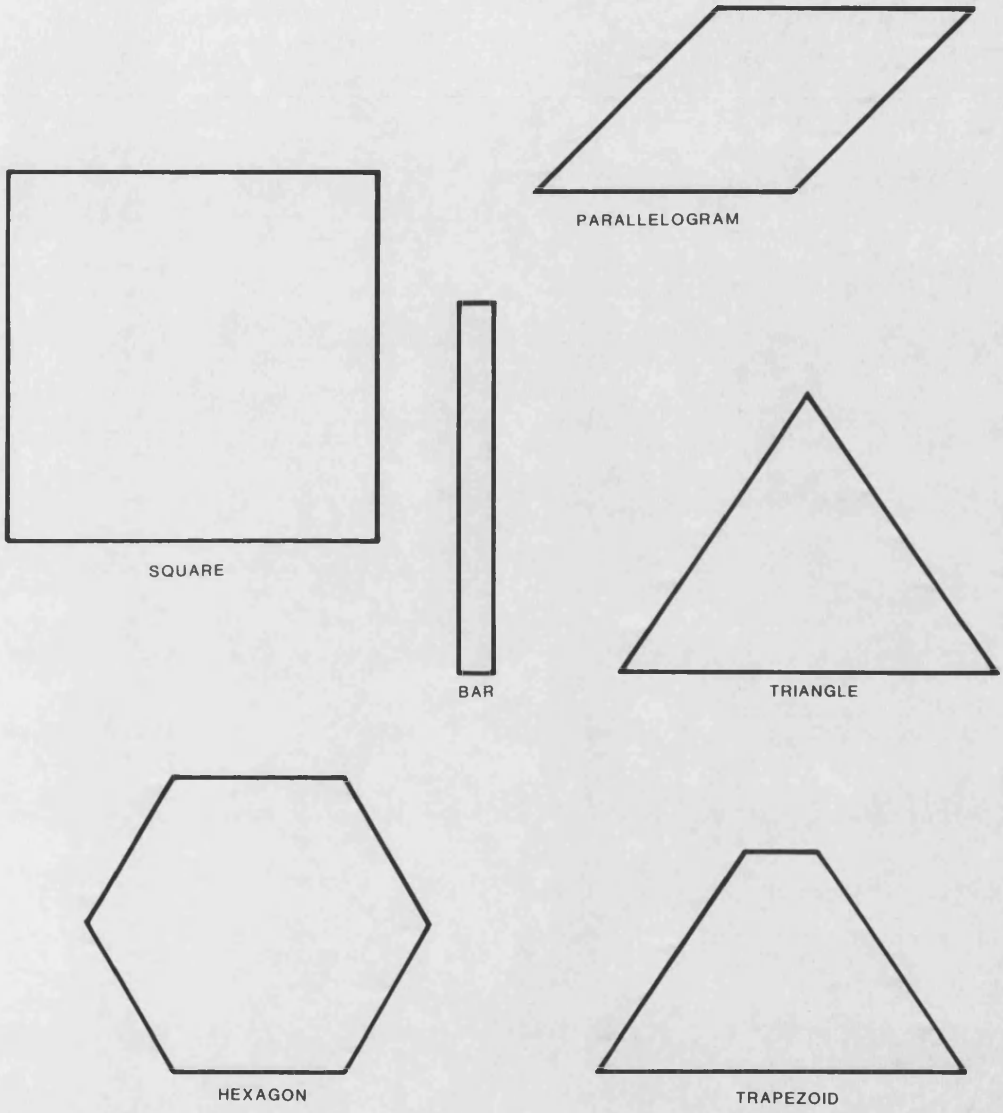
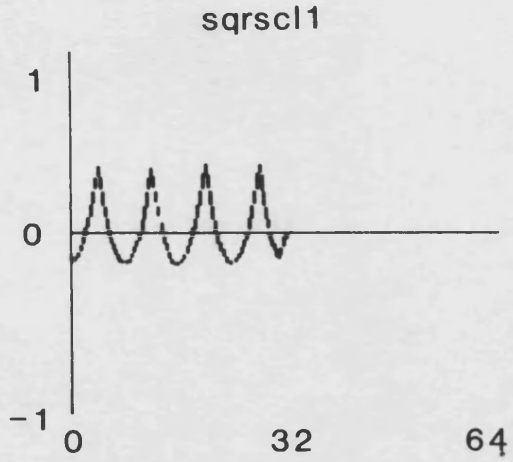
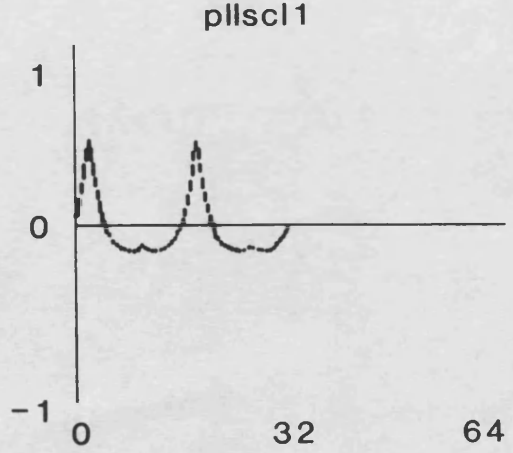


FIGURE 38 SHAPE DESCRIPTOR TEST SET

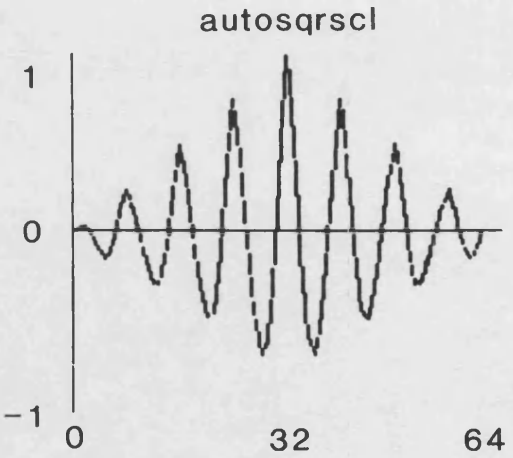
a) NORMALISED SQUARE SIGNATURE



b) NORMALISED PARALLELOGRAM SIGNATURE



c) SLIDING AUTOCORRELATION OF SQUARE



d) SLIDING AUTOCORRELATION OF PARALLELOGRAM

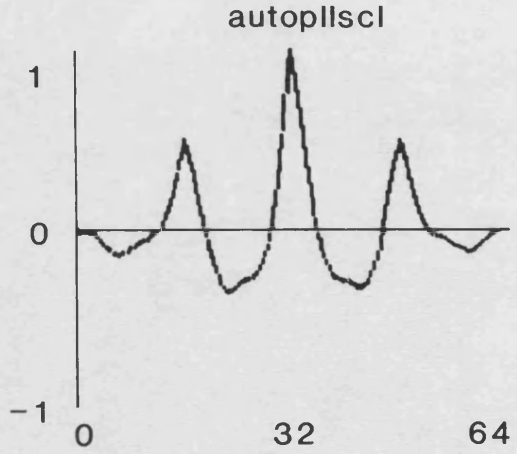
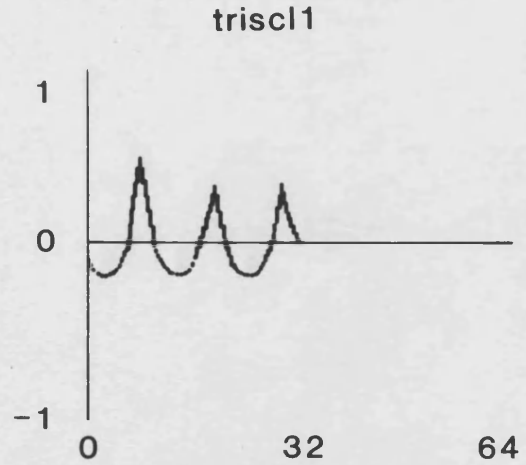
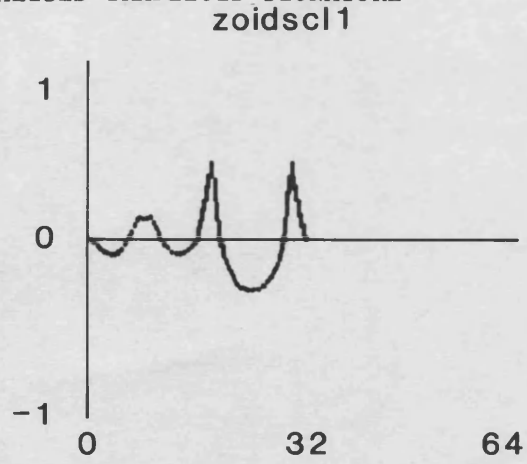


FIGURE 39

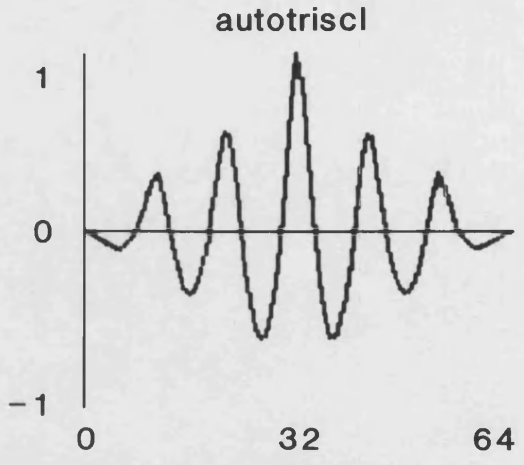
a) NORMALISED TRIANGULAR SIGNATURE



b) NORMALISED TRAPEZOID SIGNATURE



c) SLIDING AUTOCORRELATION OF TRIANGLE



d) SLIDING AUTOCORRELATION OF TRAPEZOID

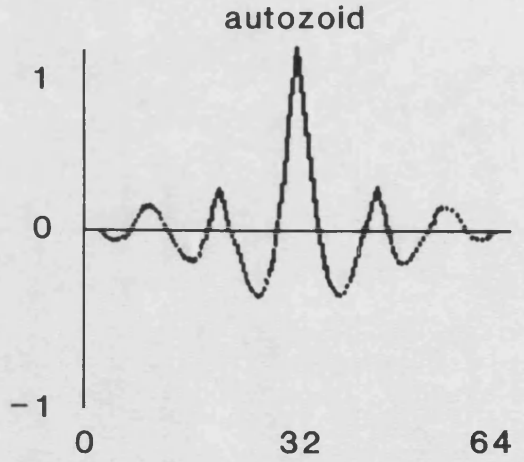
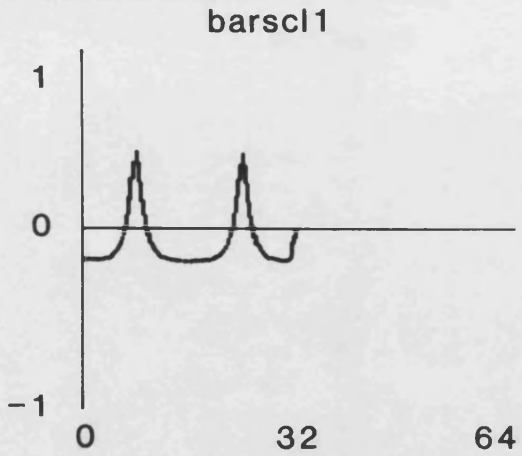
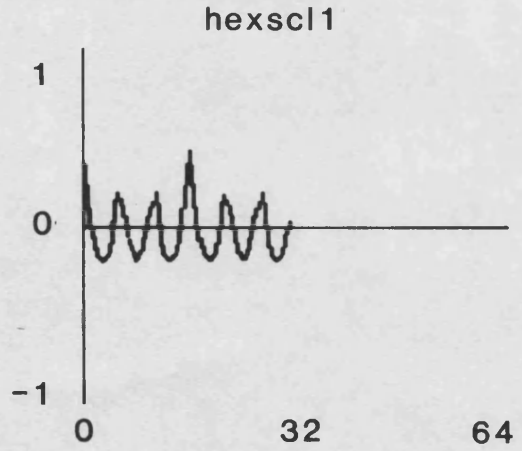


FIGURE 40

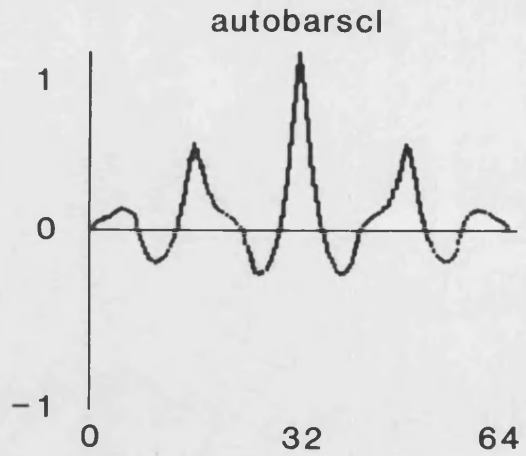
a) NORMALISED BAR SIGNATURE



b) NORMALISED HEXAGON SIGNATURE



c) SLIDING AUTOCORRELATION OF BAR



d) SLIDING AUTOCORRELATION OF HEXAGON

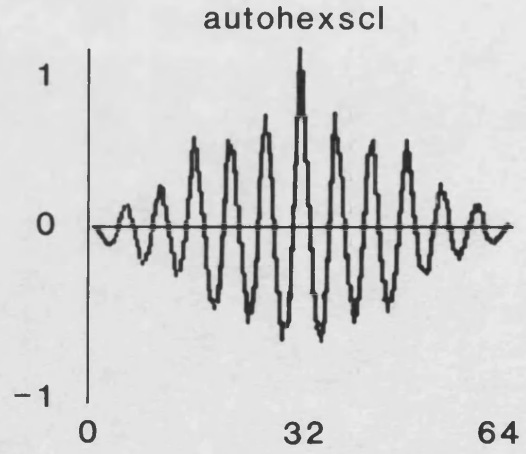
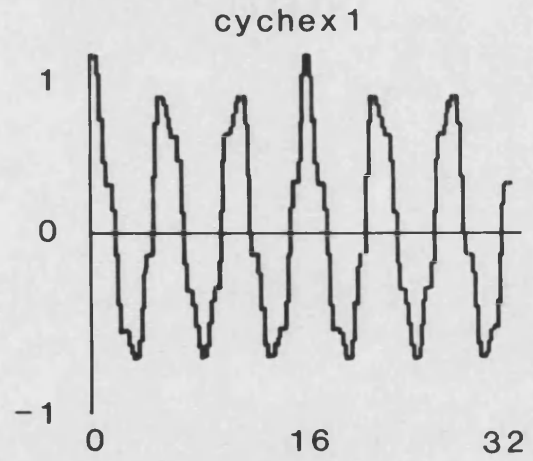
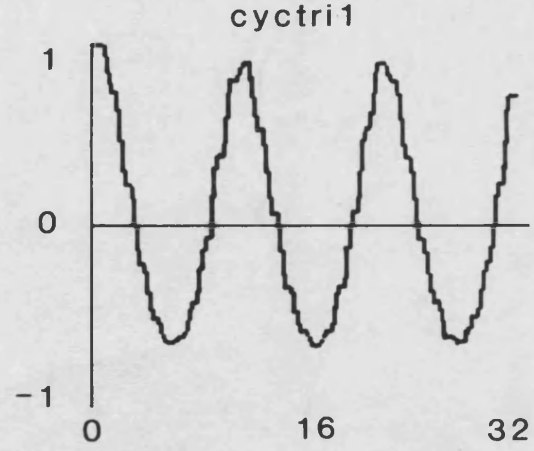


FIGURE 41

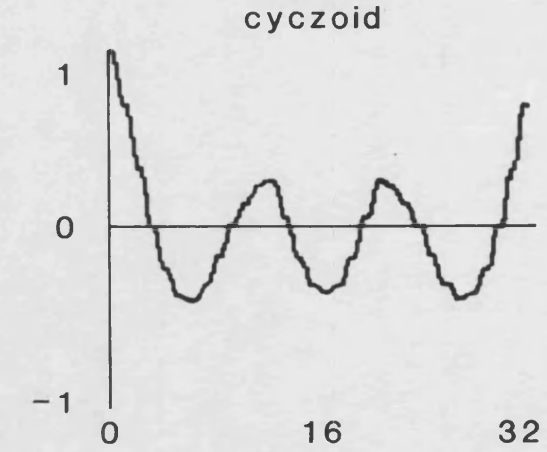
a) CYCLIC AUTOCORRELATION OF HEXAGON



b) CYCLIC AUTOCORRELATION OF TRIANGLE



c) CYCLIC AUTOCORRELATION OF TRAPEZOID



d) CYCLIC AUTOCORRELATION OF PARALLELOGRAM

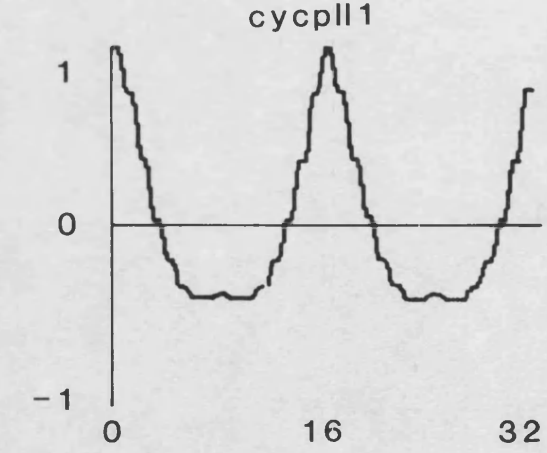
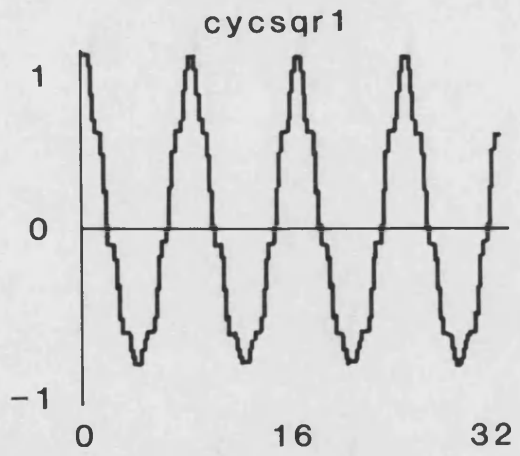
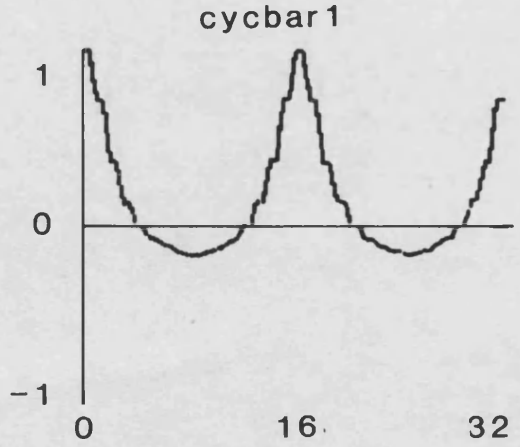


FIGURE 42

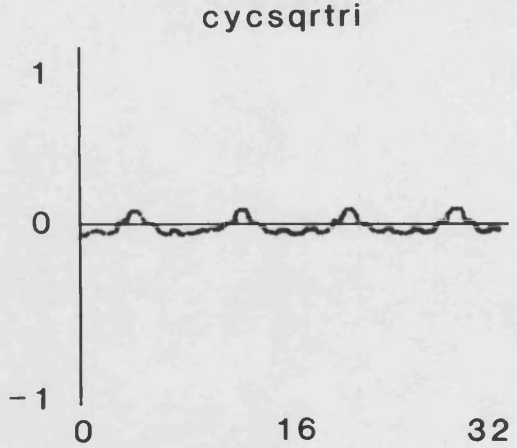
a) CYCLIC AUTOCORRELATION OF SQUARE



b) CYCLIC AUTOCORRELATION OF BAR



c) CYCLIC CROSS CORRELATION OF SQUARE/TRIANGLE



d) CYCLIC CROSS CORRELATION OF SQUARE/HEXAGON

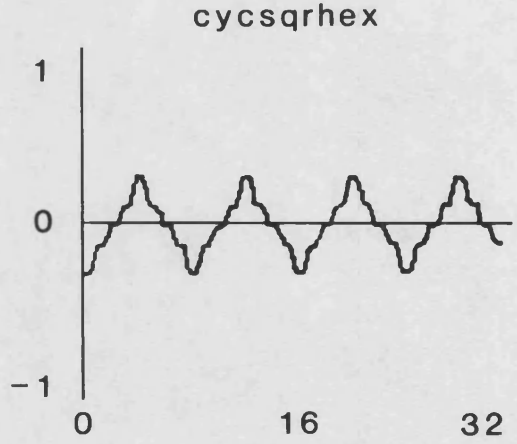
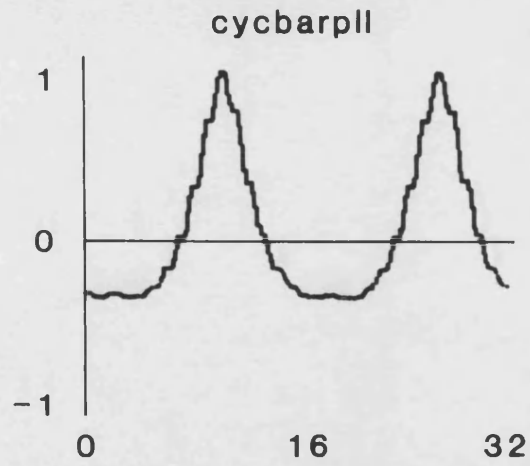
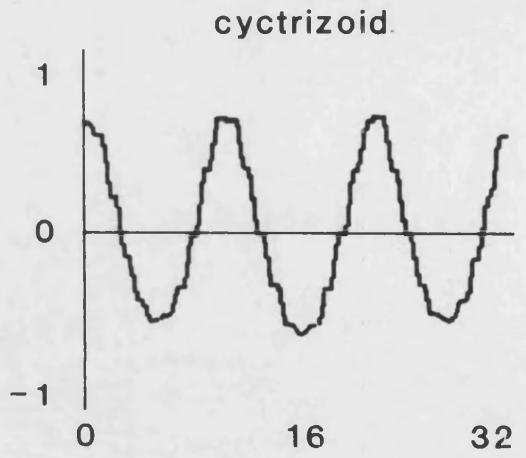


FIGURE 43

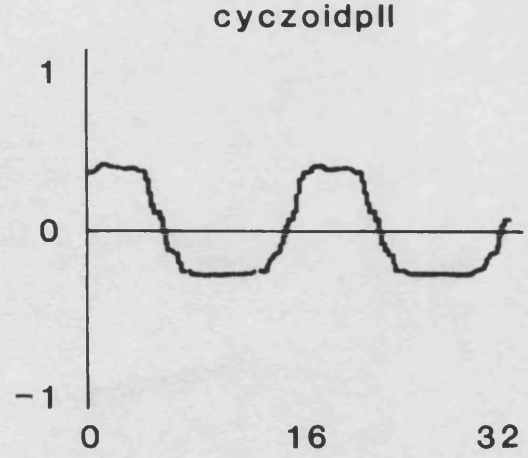
a) CYCLIC CROSS CORRELATION OF BAR/PARALLELOGRAM



c) CYCLIC CROSS CORRELATION OF TRIANGLE/TRAPEZOID



b) CYCLIC CROSS CORRELATION OF TRAPEZOID/PARALLELOGRAM



d) CYCLIC CROSS CORRELATION OF TRIANGLE/PARALLELOGRAM

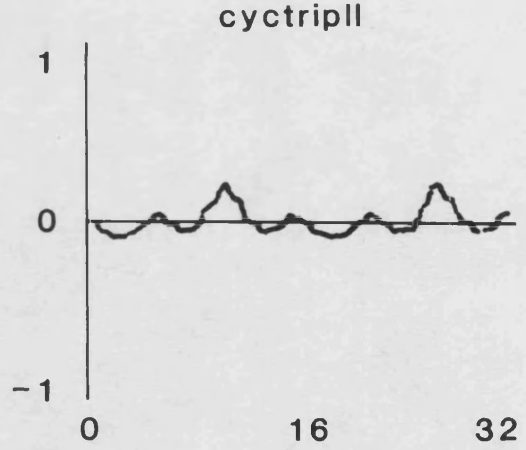
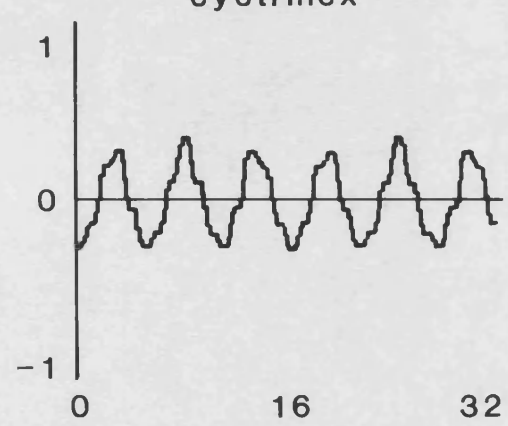
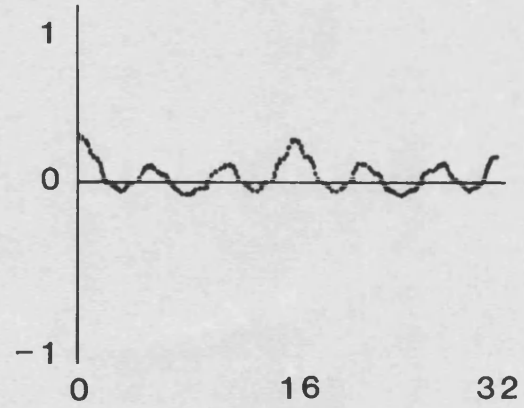


FIGURE 44

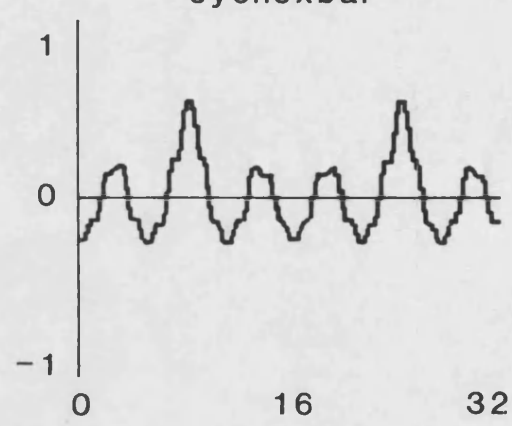
a) CYCLIC CROSS CORRELATION OF TRIANGLE/HEXAGON



b) CYCLIC CROSS CORRELATION OF TRIANGLE/BAR



c) CYCLIC CROSS CORRELATION OF HEXAGON/BAR



d) CYCLIC CROSS CORRELATION OF HEXAGON/TRAPEZOID

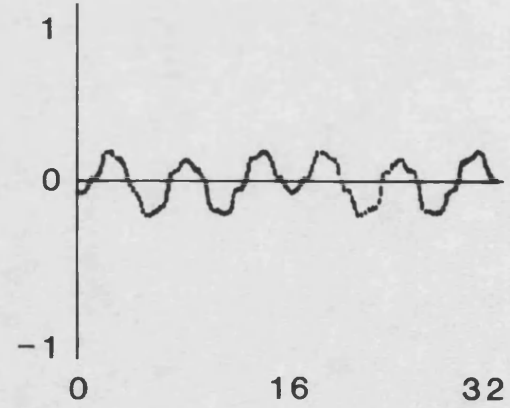
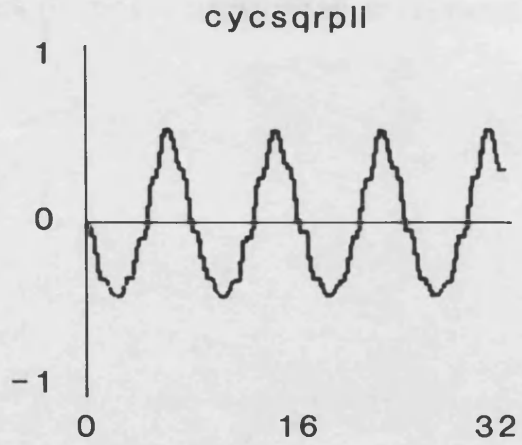
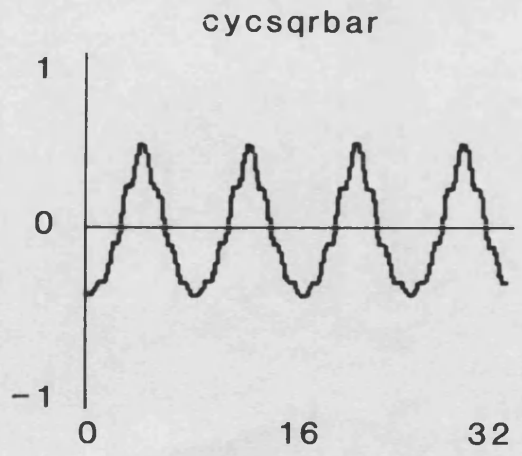


FIGURE 45

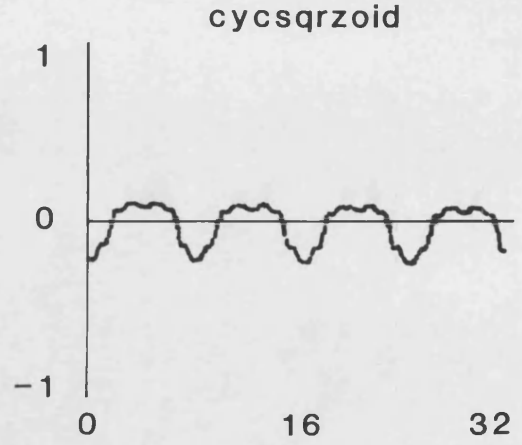
a) CYCLIC CROSS CORRELATION OF SQUARE/PARALLELOGRAM



b) CYCLIC CROSS CORRELATION OF SQUARE/BAR



c) CYCLIC CROSS CORRELATION OF SQUARE/TRAPEZOID



d) CYCLIC CROSS CORRELATION OF BAR/TRAPEZOID

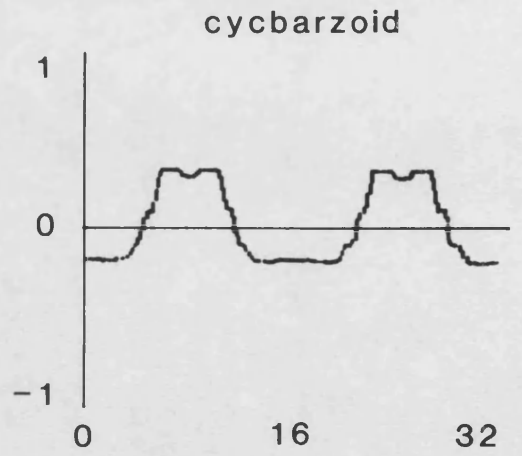


FIGURE 46

a) CYCLIC CROSS CORRELATION OF HEXAGON/PARALLELOGRAM

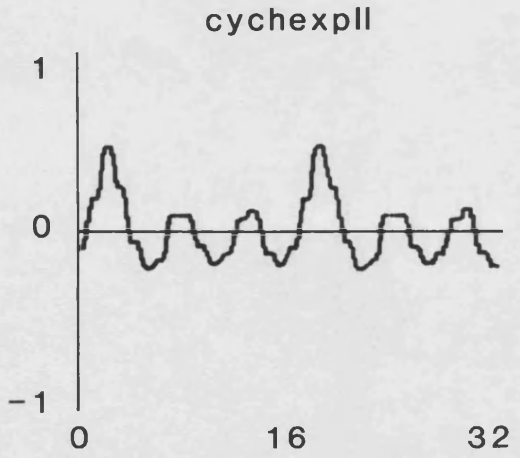


FIGURE 47

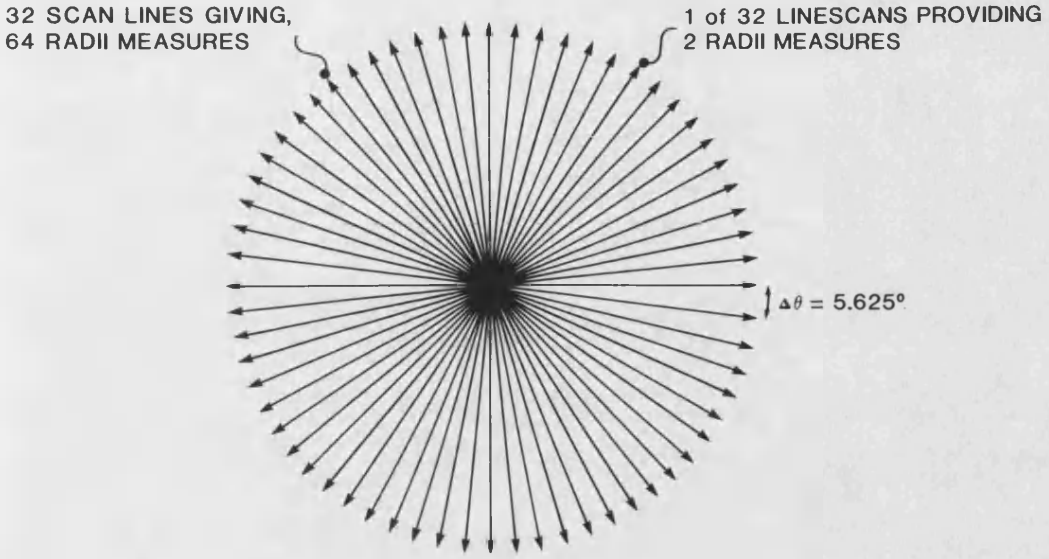
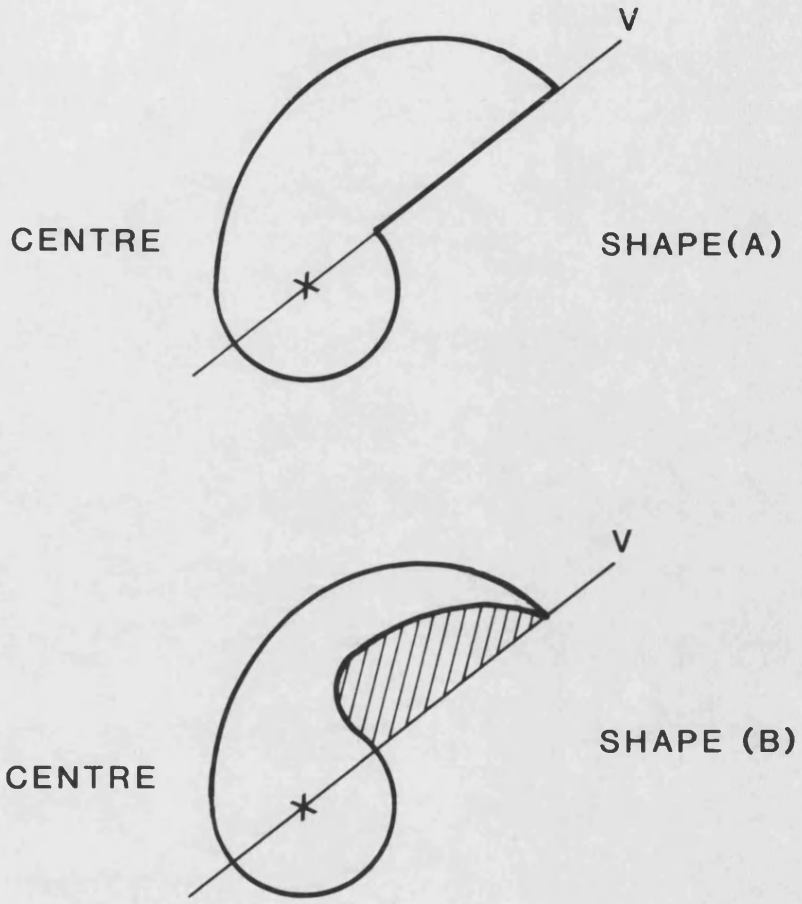
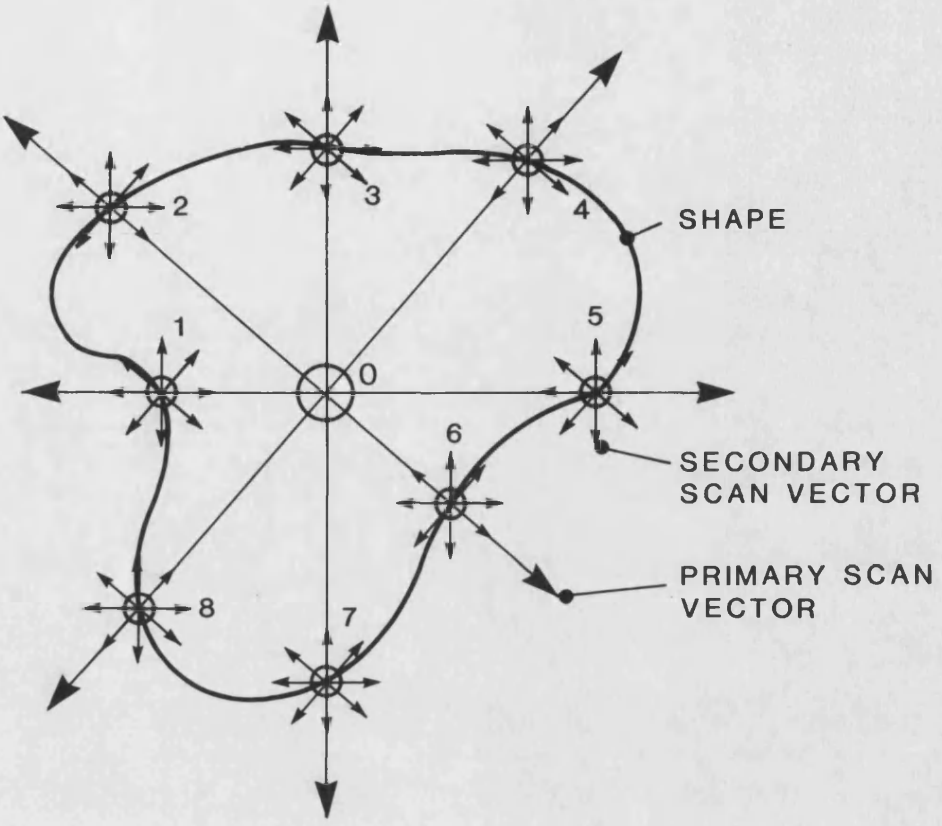


FIGURE 48 VECTORSCAN "SPOKES"



NB: Radii measurements along V for example would be similar for both shapes and hence similar boundary shape descriptions

FIGURE 49 SIMILAR SHAPES



0 = CENTROID = PRIMARY VECSCAN CENTRE

1→8 = SECONDARY SCAN CENTRES

FIGURE 50 AN ARRAY OF VECTOR SCAN NODES

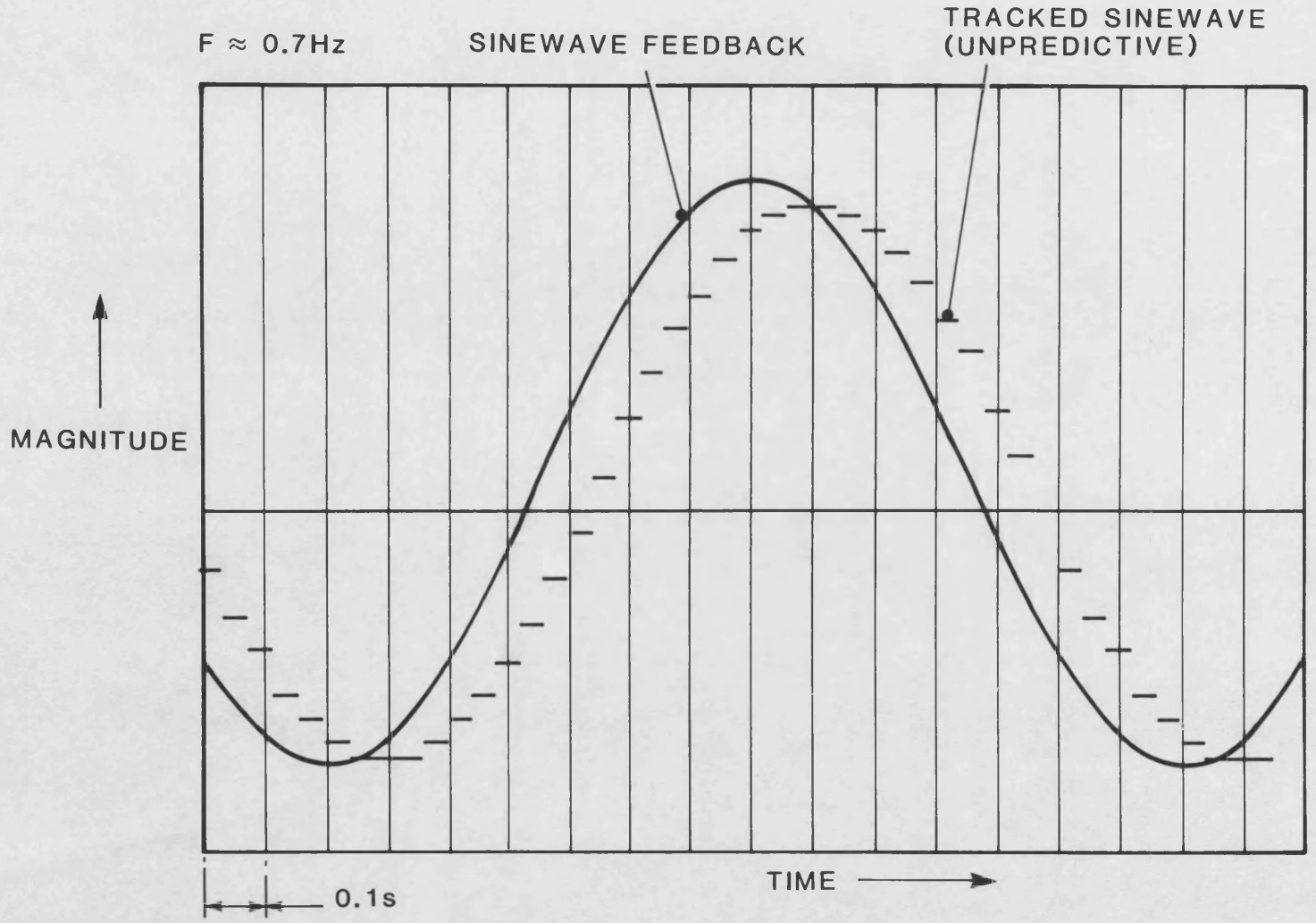
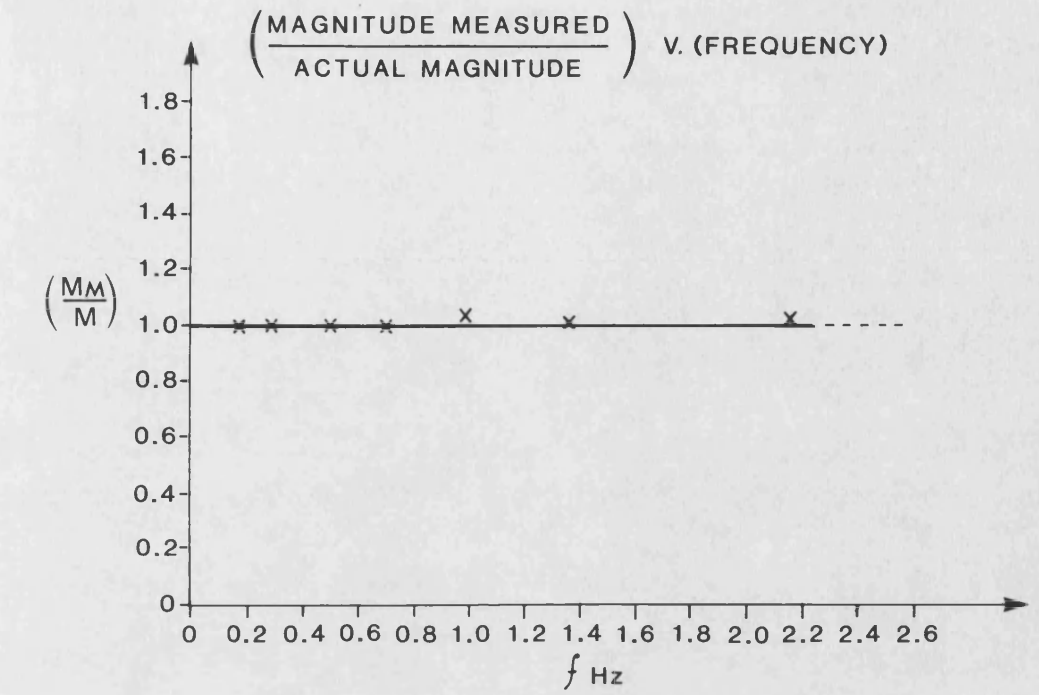


FIGURE 51 TYPICAL TRACKED TARGET POSITION TAKEN FROM VOSTTAC1 FOR A SINUSOIDAL INPUT



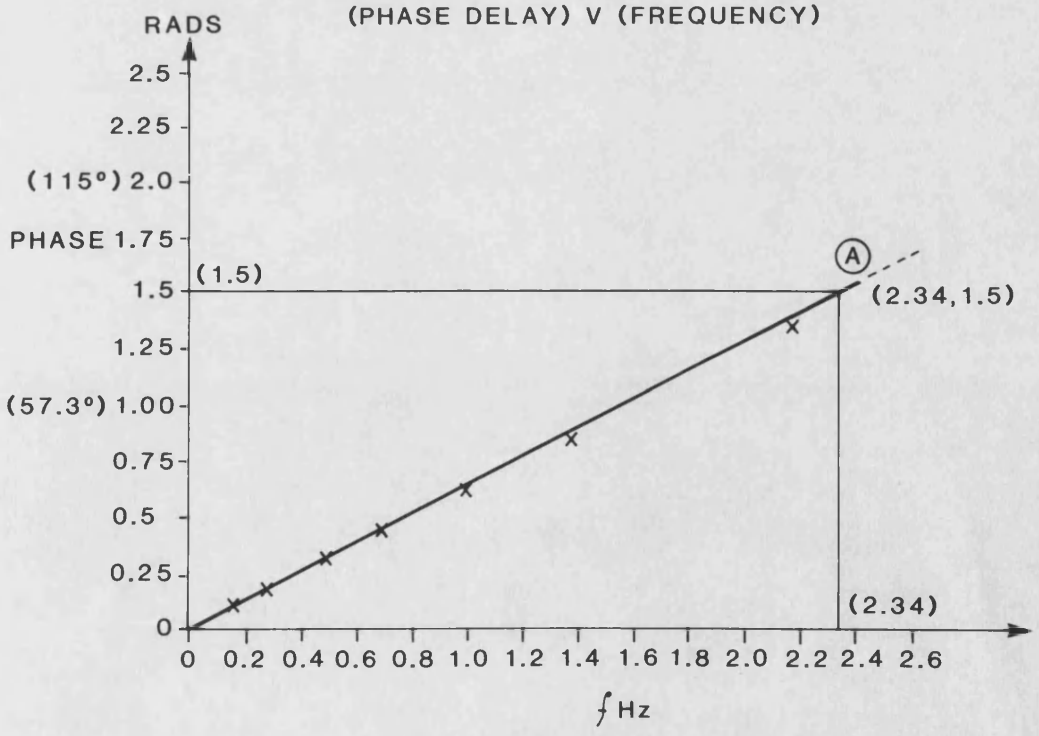
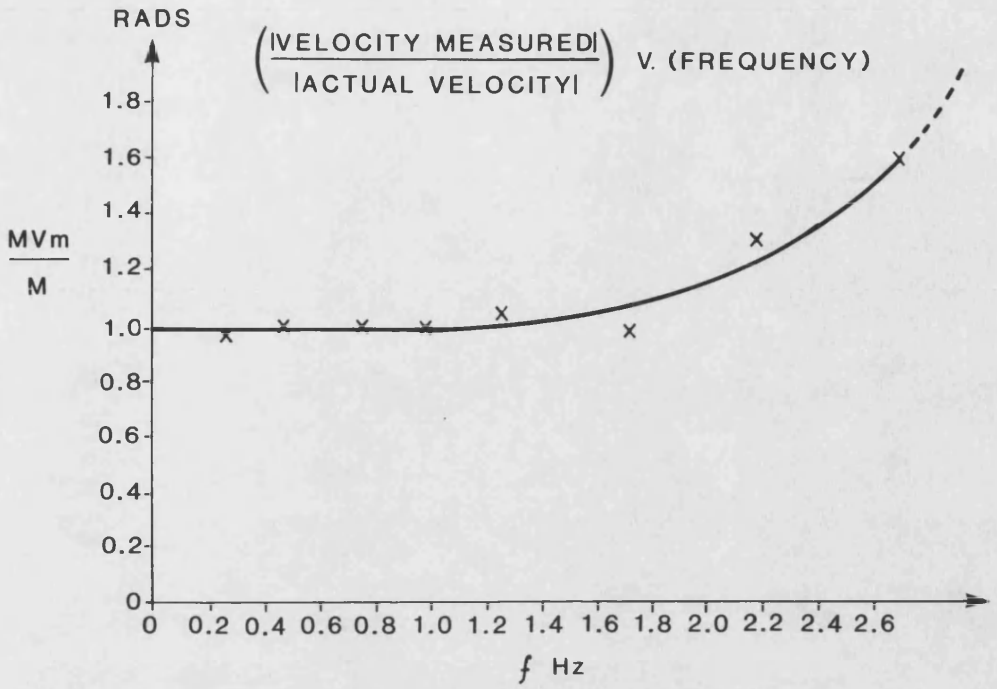


FIGURE 52 MAGNITUDE AND PHASE RESPONSE OF THE VOSTTAC1 TRACKER



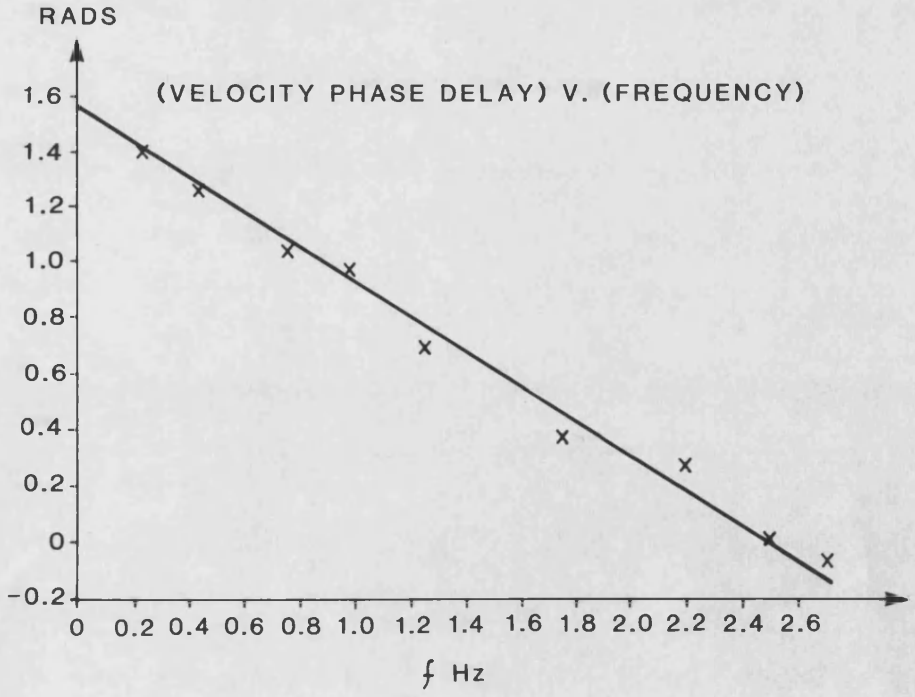
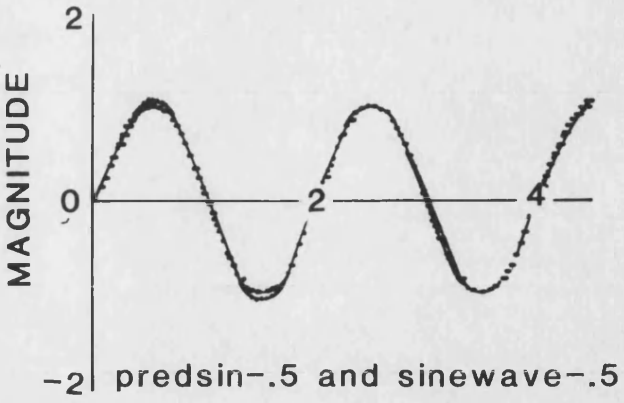
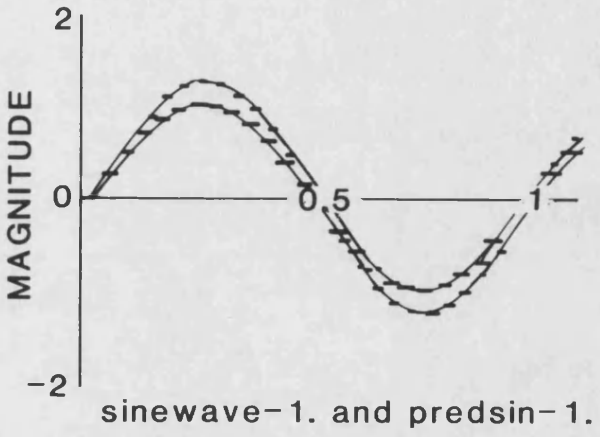


FIGURE 53

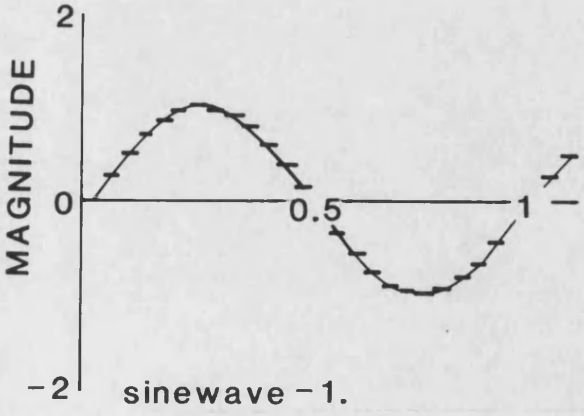
MAGNITUDE AND PHASE RESPONSE OF THE VOSTTAC1 PREDICTING TRACKER



a) ½ Hz INPUT AND PREDICTED POSITION

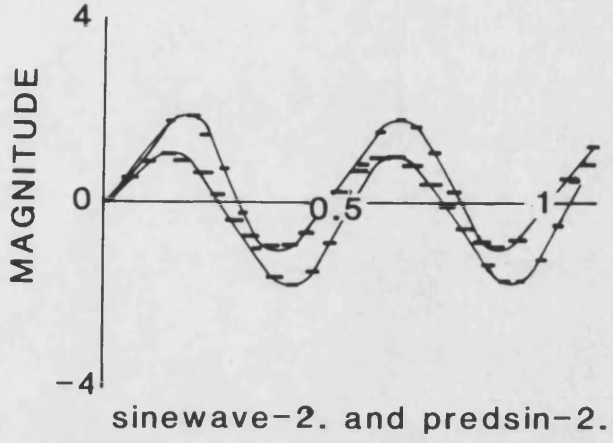


b) 1 Hz INPUT AND PREDICTED POSITION

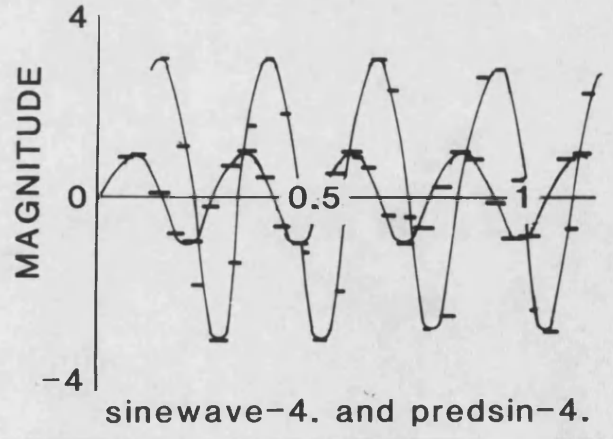


c) 1 Hz INPUT ONLY

FIGURE 54 GRAPHS OF PREDICTED AND UNPREDICTED TRACKING SIGNALS

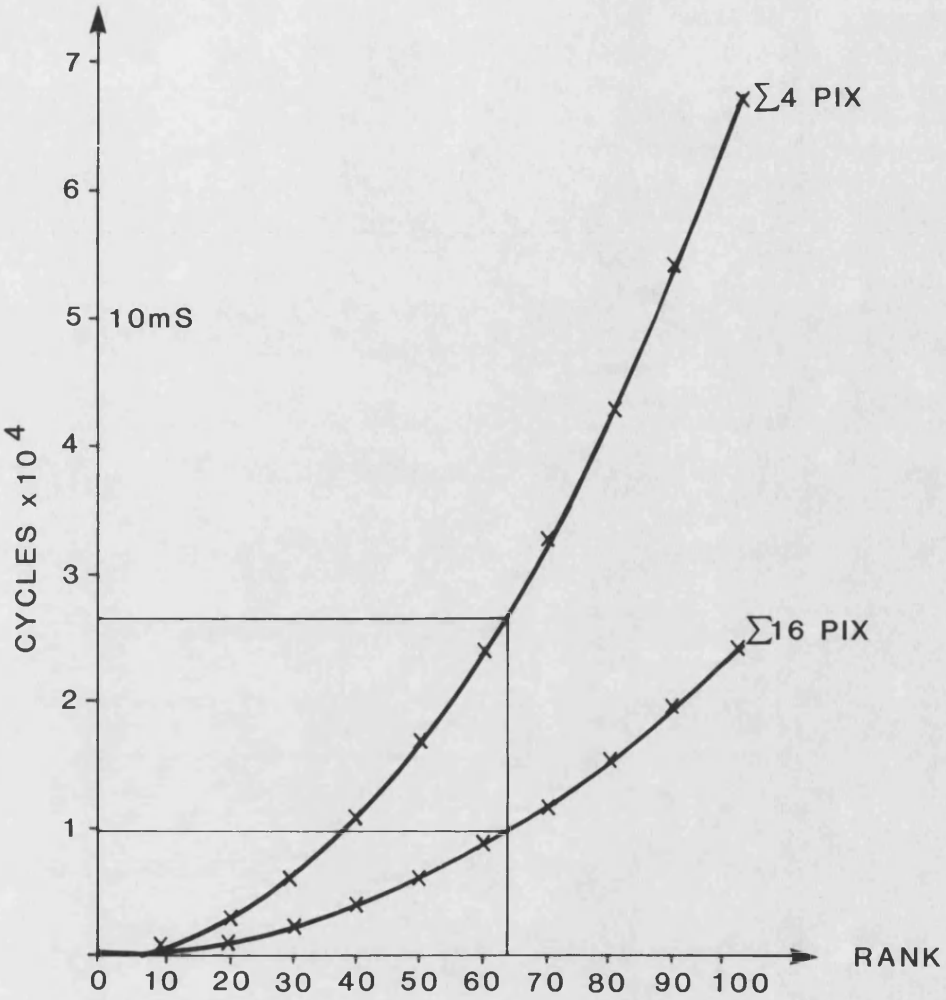


a) 2 Hz INPUT AND PREDICTED POSITION



b) 4 Hz INPUT AND PREDICTED POSITION

FIGURE 55 GRAPHS OF PREDICTED AND UNPREDICTED TRACKING SIGNALS

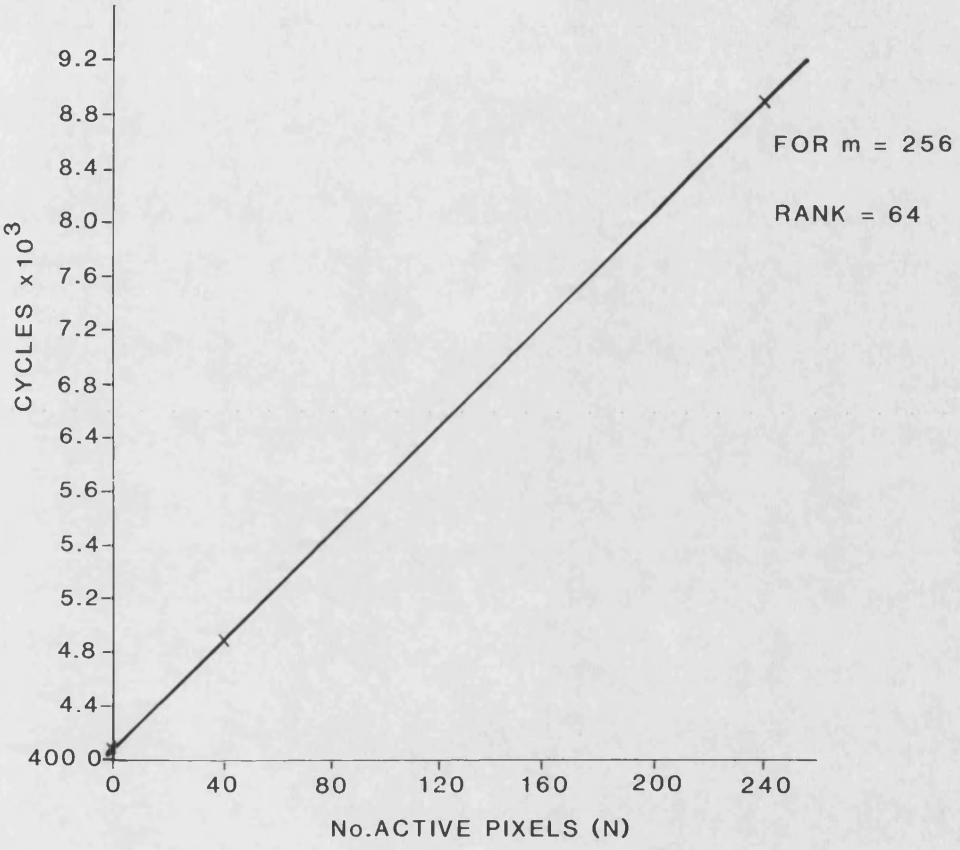


$$T\,TOTAL = T\sum 4\ PIX + T\sum 16\ PIX$$

N.B: $T\sum 4\ PIX \doteqdot RANK.\frac{RANK}{4}\ .27$

$$T\sum 16\ PIX \doteqdot \left(\frac{RANK}{4}\right)^2 .39$$

FIGURE 56 BLOCK AVERAGING CHARACTERISTIC OF SEGMENTER

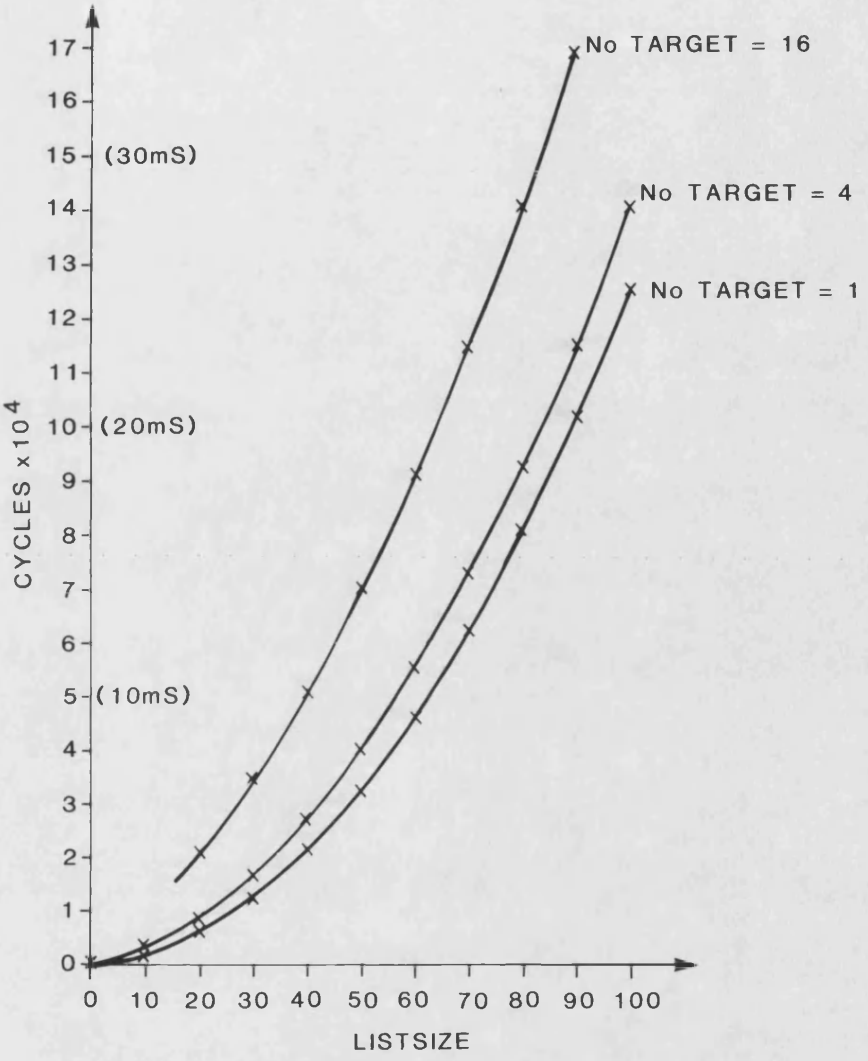


T TOTAL ≈ N*20 + mx16 CYCLES

WHERE N = NUMBER OF ACTIVE PIXEL BLOCKS

$$m = \text{NUMBER OF PIXEL BLOCKS} = \left(\frac{\text{RANK}}{4}\right)^2$$

FIGURE 57 PRIMARY GROUP ASSIGNMENT CHARACTERISTIC OF SEGMENTER



T TOTAL $\approx$ LISTSIZE x 25 x (2 x No TARGETS) + (LISTSIZE)^2 x 12 (CYCLES)

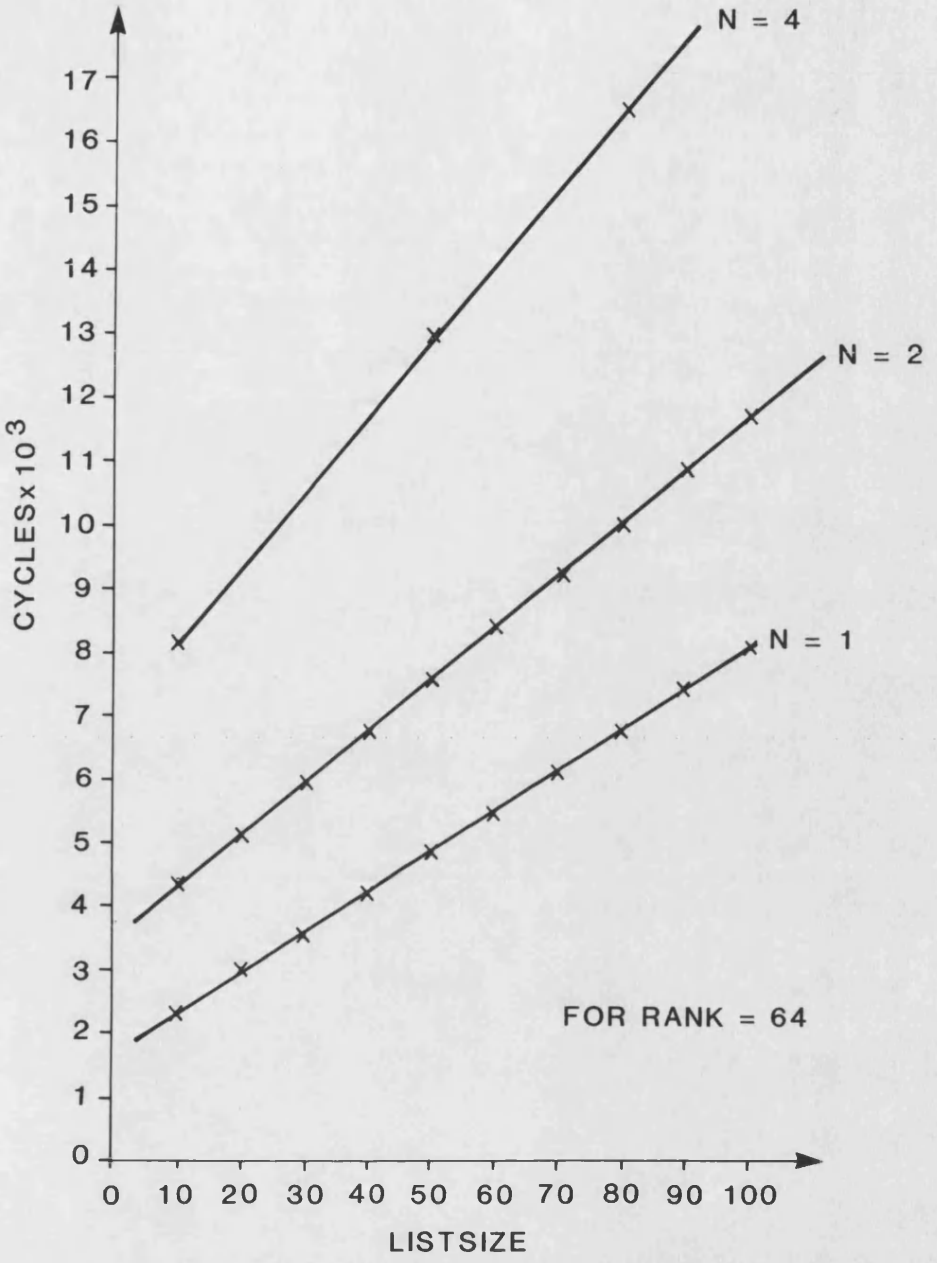
$\approx$ LISTSIZE x (50 x No TARGETS + 12 x LISTSIZE) (CYCLES)

N.B: LISTSIZE IS NUMBER OF ACTIVE BLOCKS

SPEED MUCH MORE SENSITIVE TO ACTIVE BLOCKS (SET AREA)

THAN TO NUMBER OF TARGETS (1 CYCLE $\doteq$ 200.10^{-9} S)

FIGURE 58 SECONDARY GROUP ASSIGNMENT CHARACTERISTIC OF SEGMENTER



$$TTOTAL = N \times \left(75 + \text{"LIST SIZE"}\left(19 + \frac{45}{N}\right) + RANK^2 \times \frac{13}{32}\right) \text{ CYCLES}$$

N = No TARGETS
EXAMPLES
LISTCOUNT = 100
RANK = 64

		CYCLES	mS
N = 1	TOTAL =	$8.139.10^3$	1.63
2		$1.178.10^4$	2.36
4		$1.906.10^4$	3.81
8		$3.361.10^4$	6.72
16		$6.272.10^4$	12.54
32		$1.209.10^5$	24.42
64		$2.374.10^5$	47.48

FIGURE 59 MULTI-TARGET CENTRE/AREA FINDING FOR VARIABLE LIST SIZE

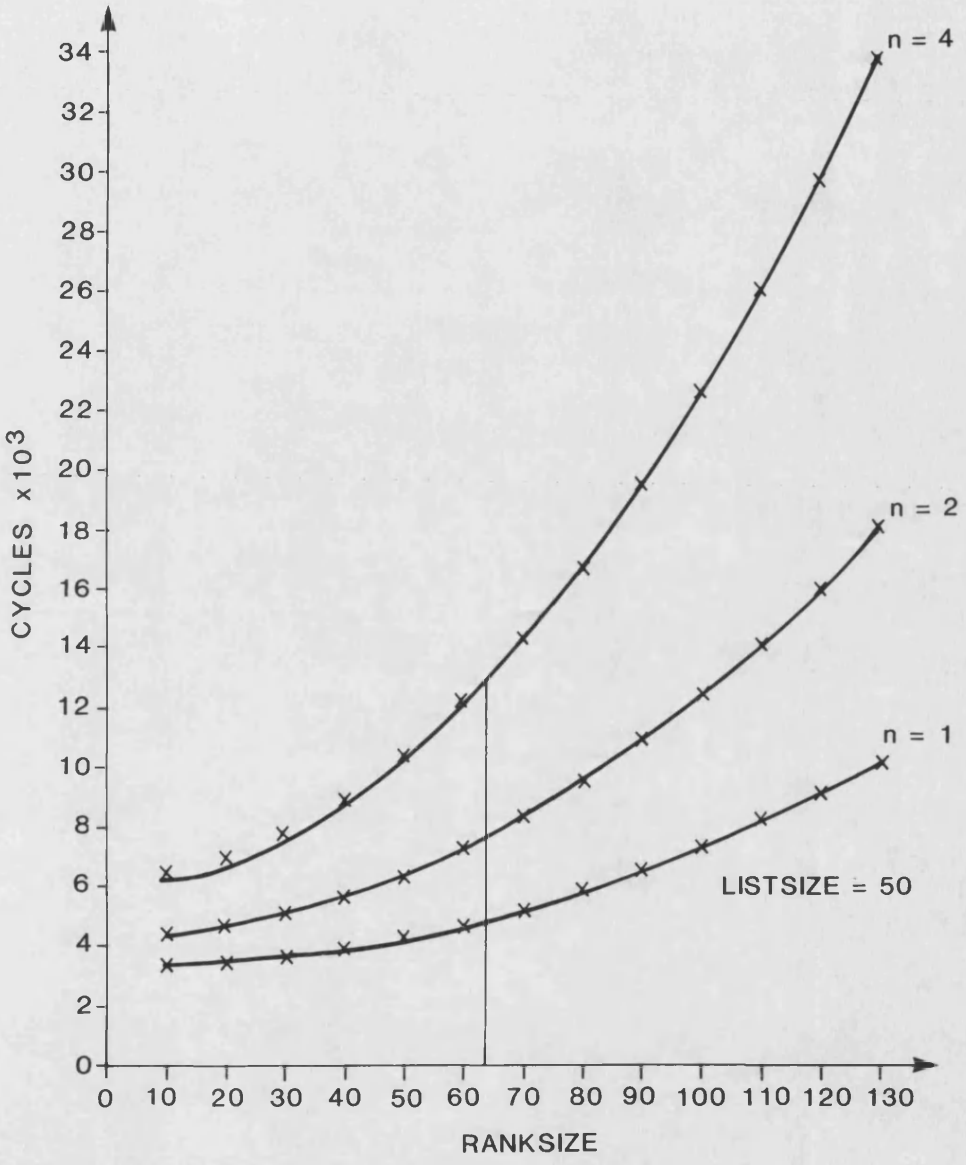


FIGURE 60 MULTI-TARGET CENTRE/AREA FINDING FOR VARIABLE WINDOW RANK

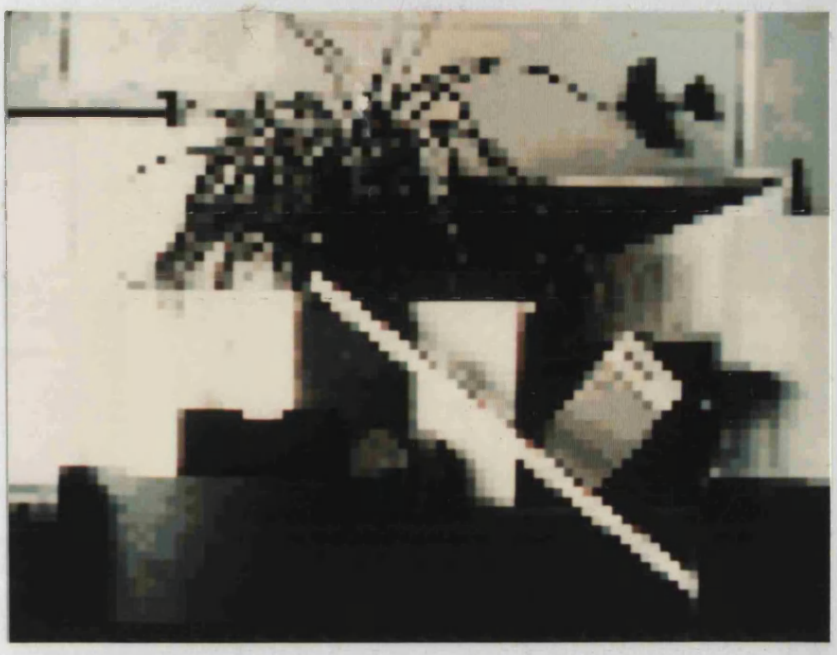
db/km	λ 0.7μm	λ 1.06μm	λ 10.6μm	
0.2	EXTREMELY CLEAR	EXTREMELY CLEAR	SUB ARTIC WINTER	
0.6	STANDARD CLEAR (v = 25km)	STANDARD CLEAR	CLEAR	
0.8		CLEAR		
1.0	CLEAR (v = 15km)		HAZE SUB ARTIC SUMMER	
2.0	LIGHT HAZE (v = 15km)		MID LATITUDE SUMMER	RAIN 2.5mm/h
3.0	MEDIUM HAZE (v = 5km)		TROPICAL	RADIATION FOG v = 0.5km RAIN 4mm/h
5.0	HAZE (v = 3km)			SNOW 0.2mm/h

v = visibility

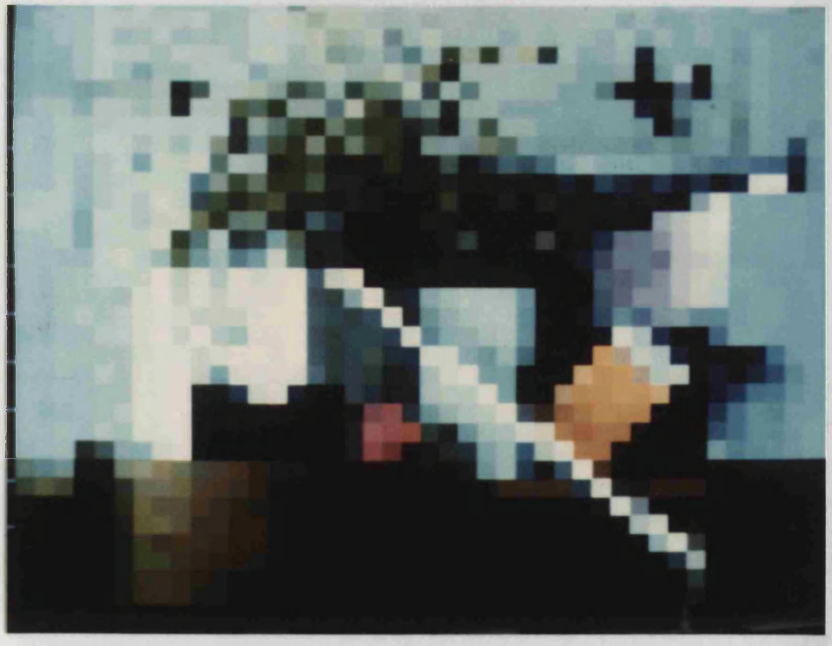
FIGURE 61 ATMOSPHERIC ATTENUATION OF SIGNAL WAVELENGTHS.

PIC1 B/W test image 360*577 resolution

PIC2 B/W test image 180*144 resolution



PIC3 B/W test image 90*72 resolution

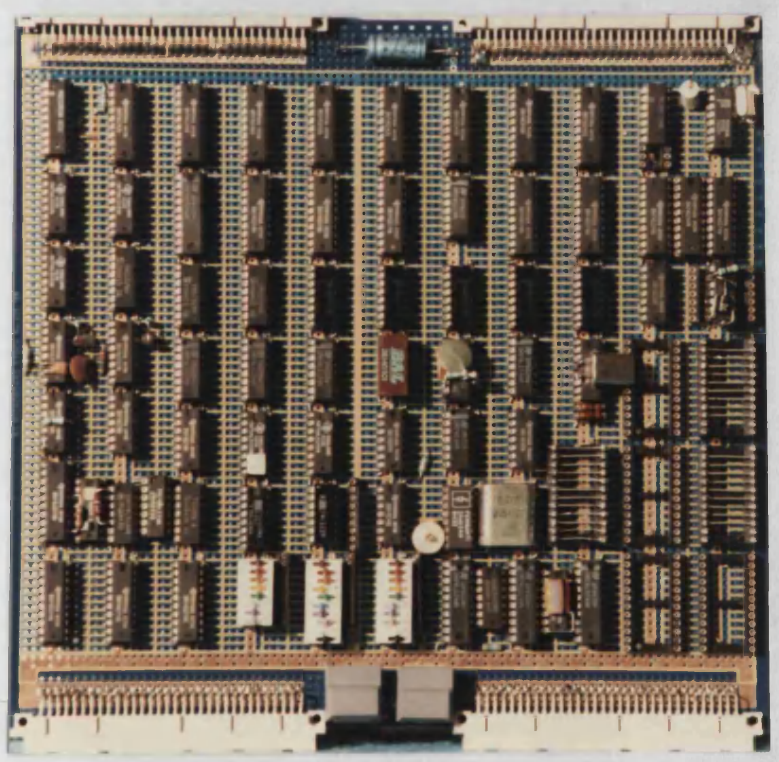


PIC4 Colour test image 45*36 resolution

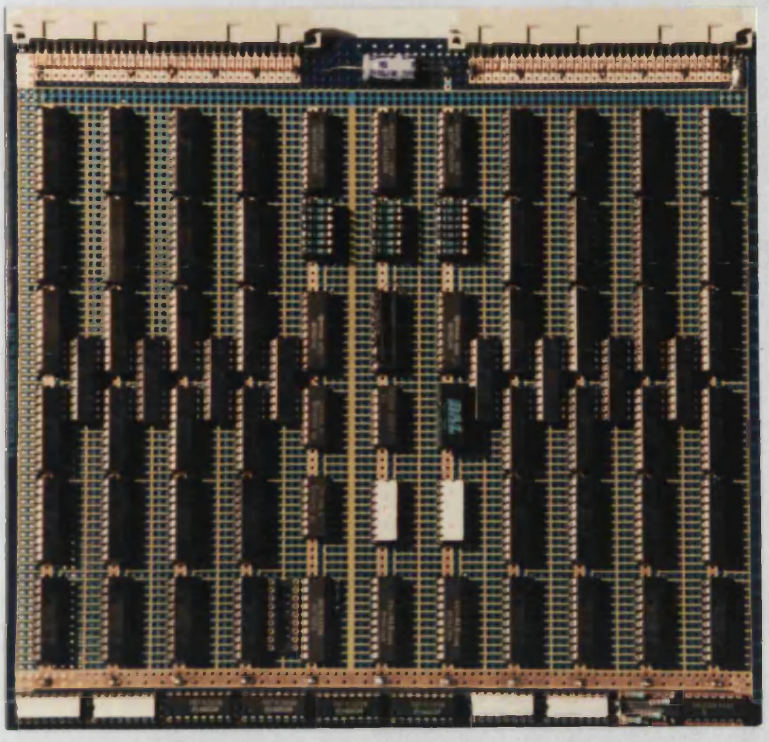
PIC5 Colour image with 512 possible colour combinations

PIC6 Colour image with 64 possible colour combinations

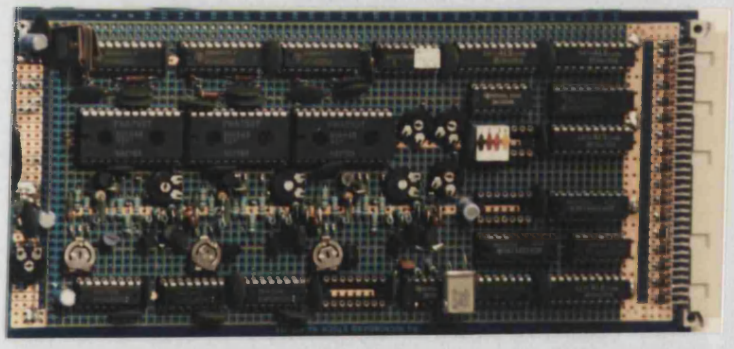
PIC7 Colour image with 8 possible colour combinations



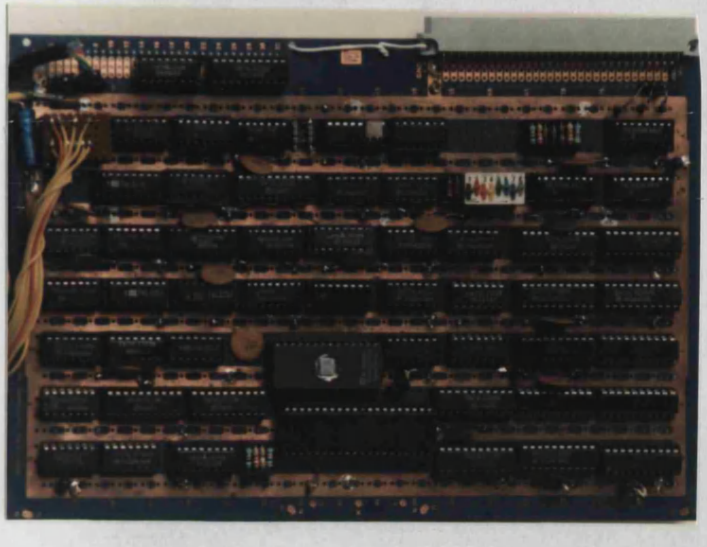
PIC8 Video Scanner Board
(1 per markI system)



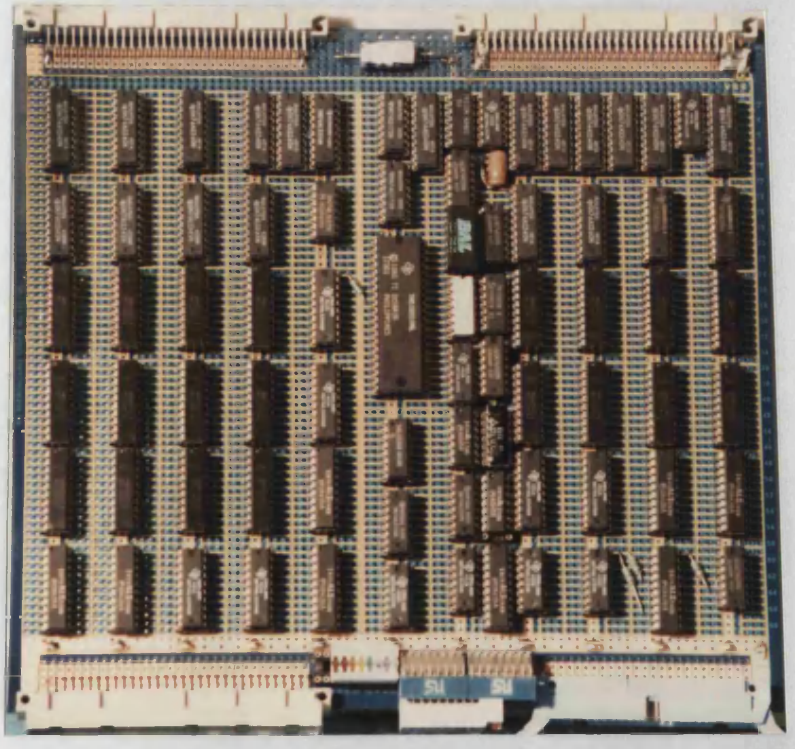
PIC9 Framestore Board including a DTP
(6 per markI system)



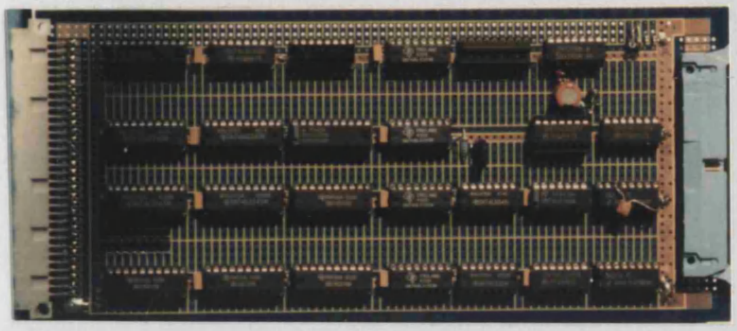
PIC10 Video Digitizer/Amplifier Board
(1 per markI system)



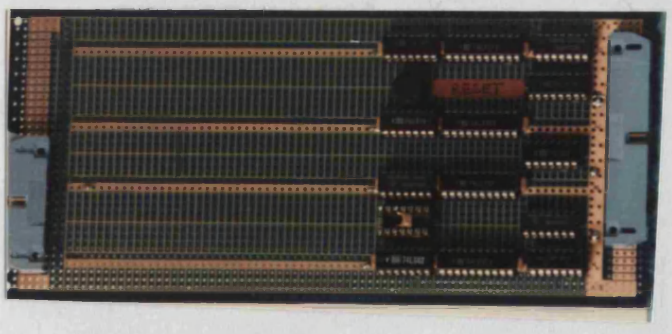
PIC11 Target Nomination and Auxiliary Display Unit
(1 per markI system)



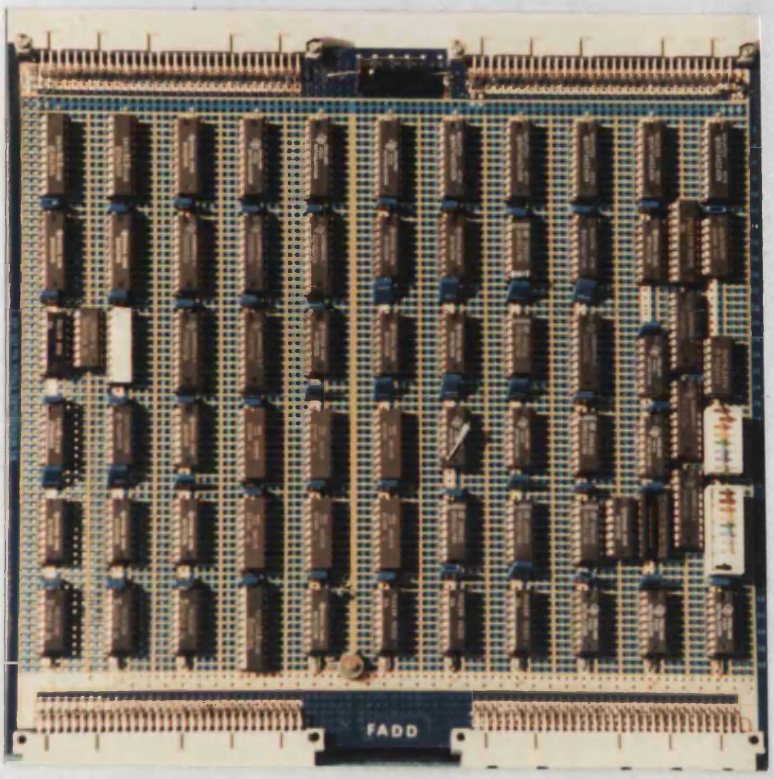
PIC12 TMS32010 Processor Board
(3 to 5 in markI system)



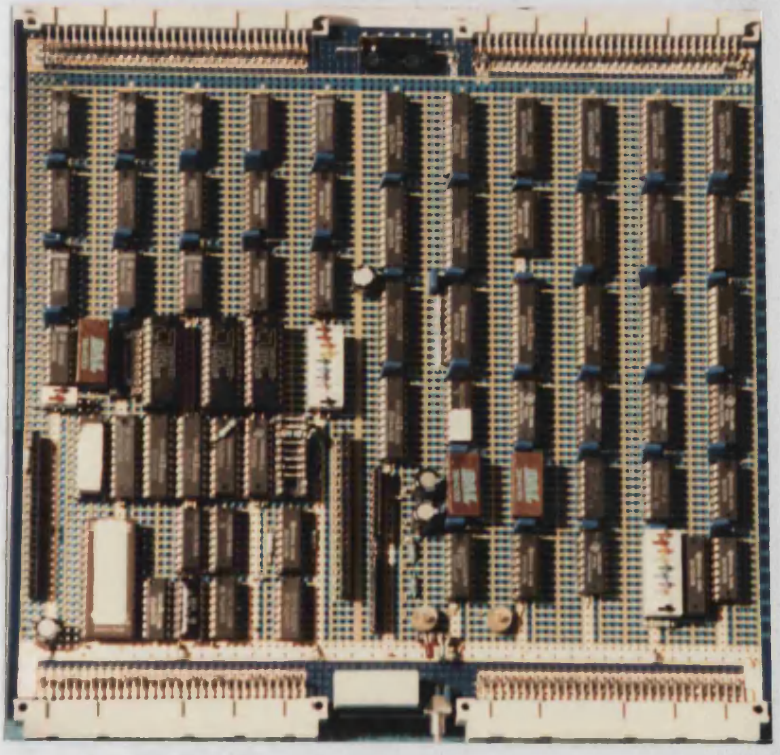
PIC13 Processor Expansion Board
(1 to 5 in markI system)



PIC14 BBC Model B Dump/Load Interface Board
(1 per markI system)

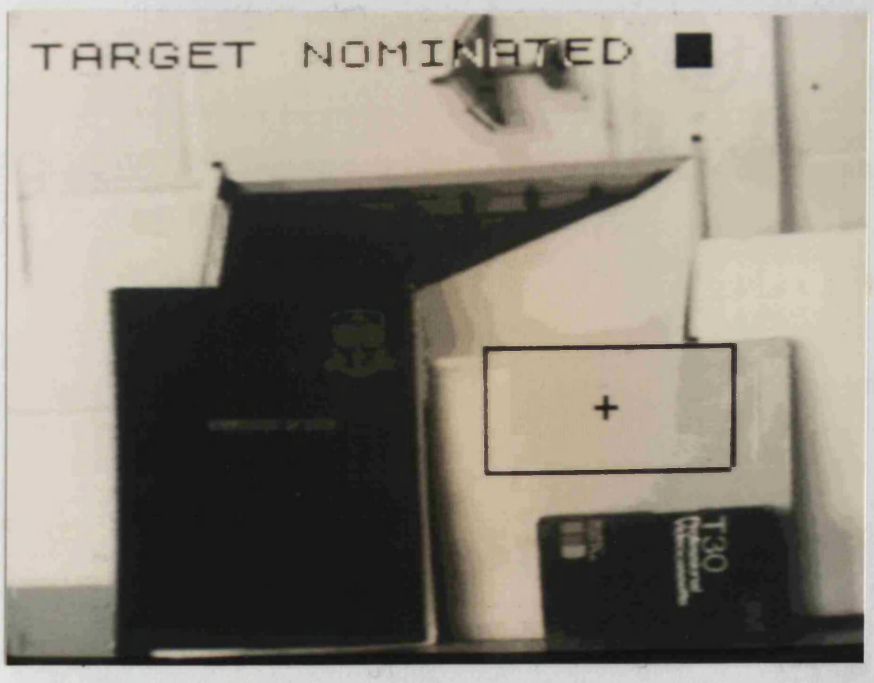


PIC15 Frame Addresser Board
(1 per markI system)



PIC16 Window Addresser Board
(1 per markI system)

PIC17 Low contrast test image

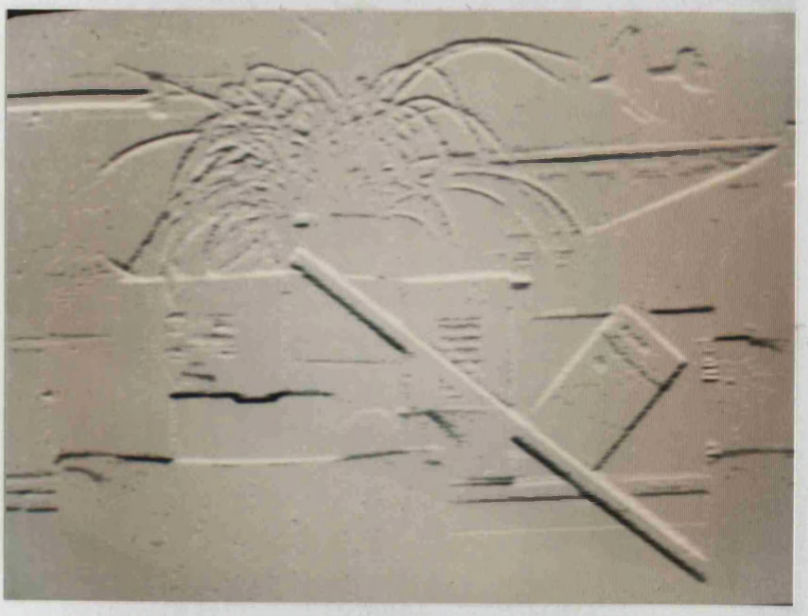


PIC18 Contrast enhanced version of PIC17

PIC19 Cluttered B/W test image

PIC20 Convolution with 3*3 Laplacian mask
(linear output)

PIC21 Convolution with 3*3 Vertical Kirsch/Prewitt mask (linear output)



PIC22 Convolution with 3*3 Horizontal Kirsch/Prewitt mask (linear output)

PIC23 Convolution with 3*3 Vertical Sobel mask
(linear output)

PIC24 Convolution with 3*3 Horizontal Sobel mask
(linear output)

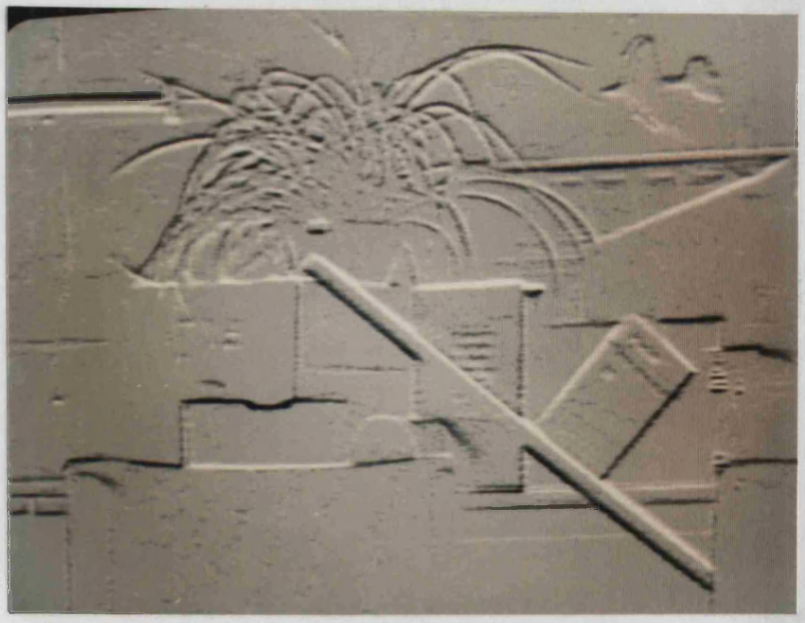
PIC25 Convolution with 2*2 45° Roberts mask
(linear output)

PIC26 Convolution with 2*2 -45° Roberts mask
(linear output)

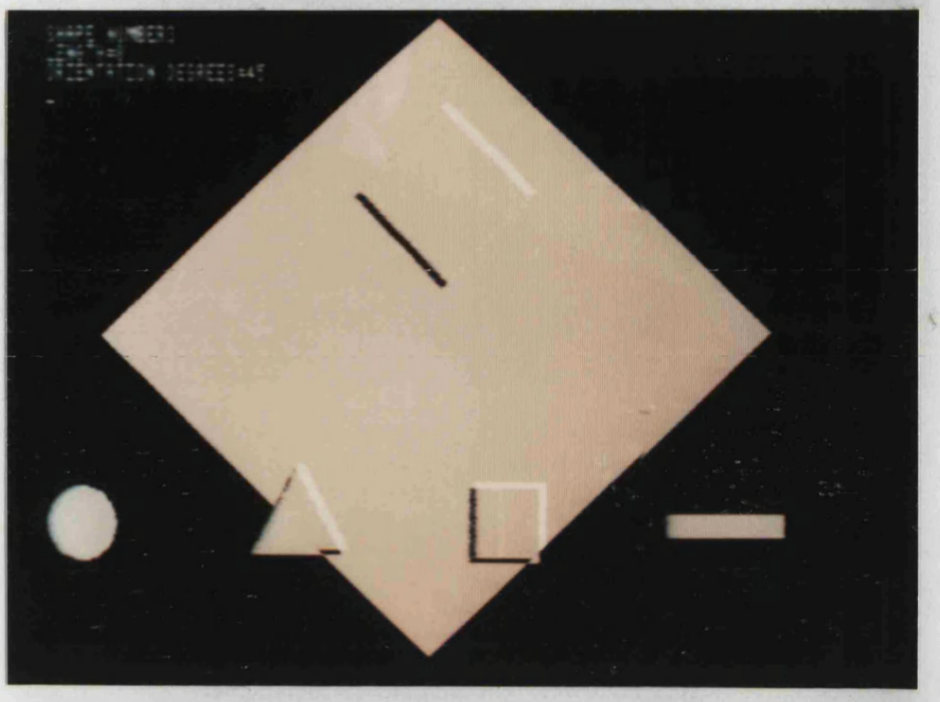
PIC27 Convolution with 3*3 East Gradient mask
(linear output)

PIC28 Convolution with 3*3 West Gradient mask
(linear output)

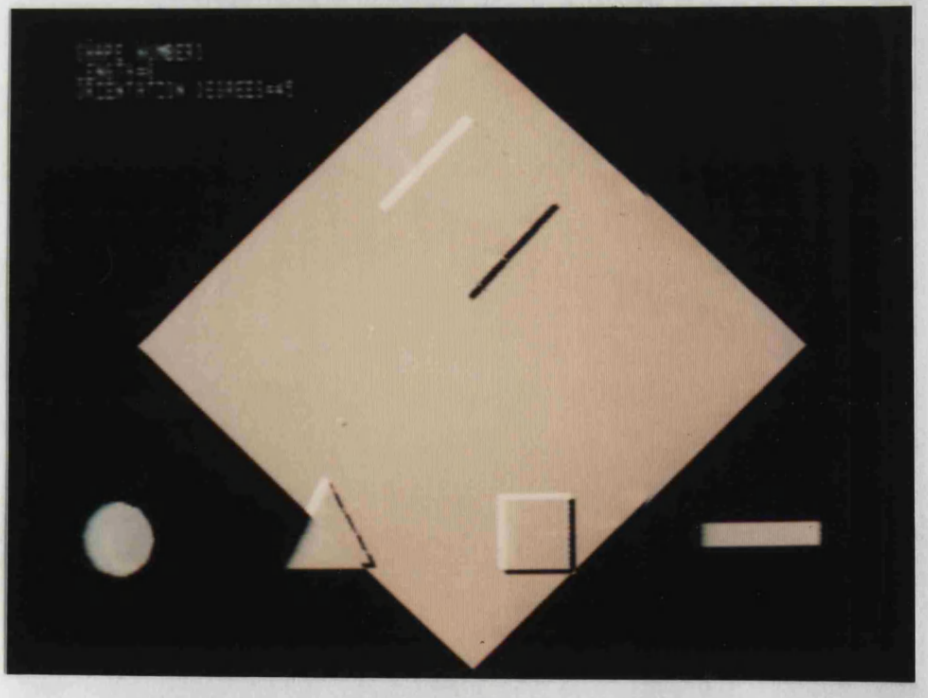
PIC29 Convolution with 3*3 North Gradient mask
(linear output)



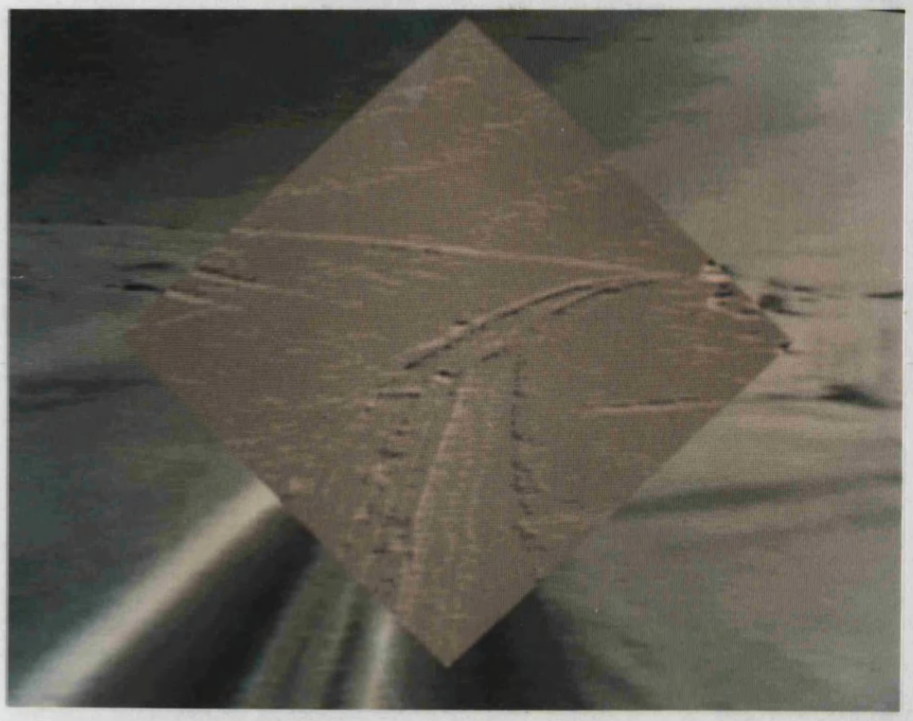
PIC30 Convolution with 3*3 South Gradient mask
(linear output)



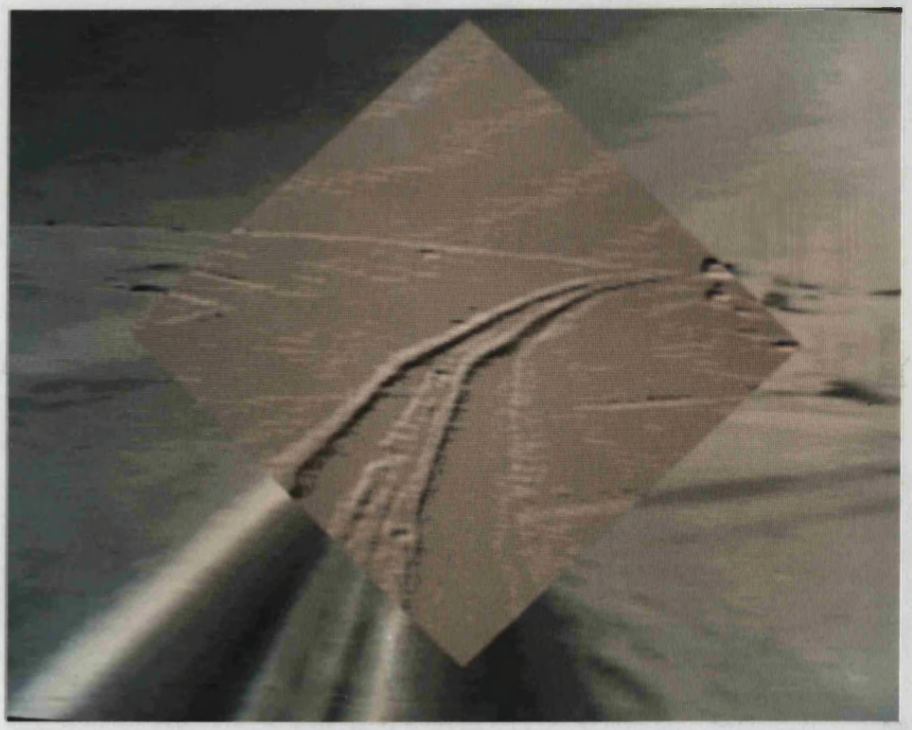
PIC31 Vector Scanning 45° edge enhancement
(linear output)



PIC32 Vector Scanning -45° edge enhancement
(linear output)



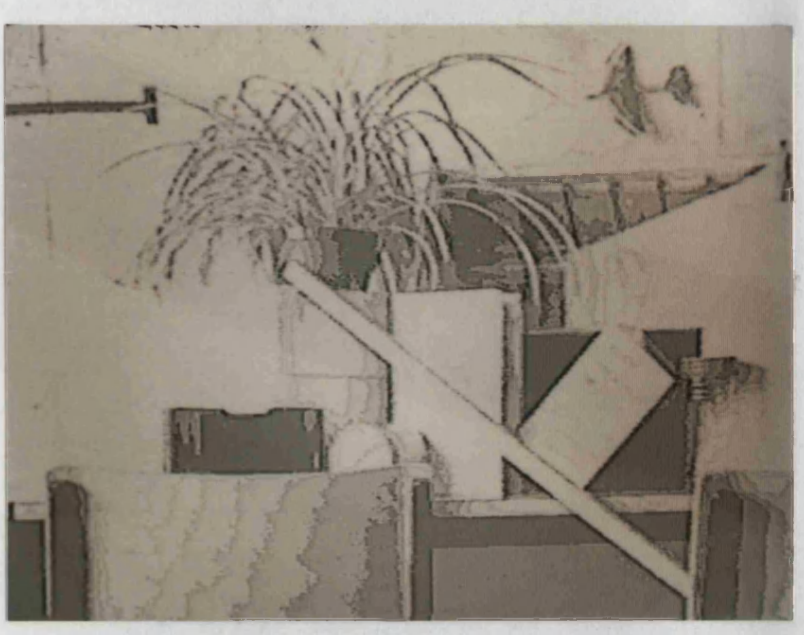
PIC33 Vector Scanning 45° edge enhancement of snowtracks (linear output)



PIC34 Vector Scanning -45° edge enhancement of snowtracks (linear output)

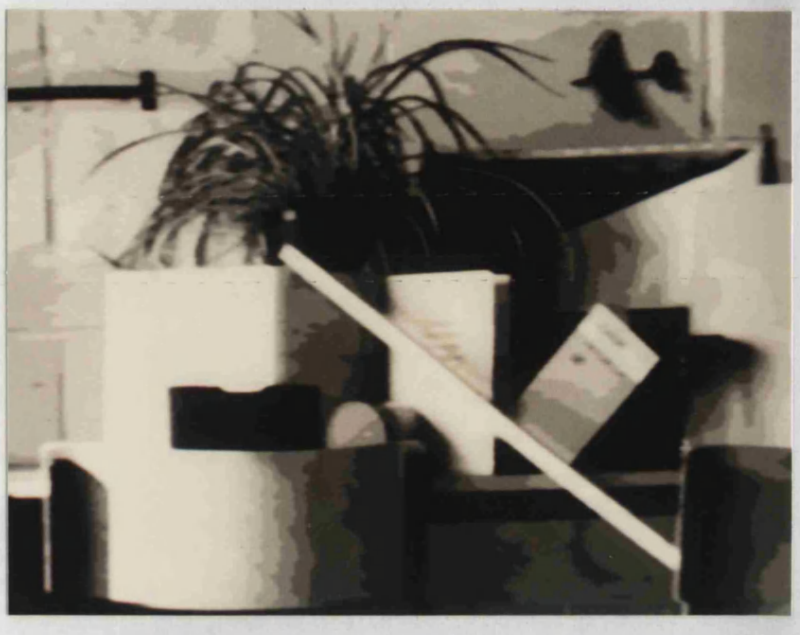
PIC35 Convolution with 3*3 vertical line segment mask
(linear output)

PIC36 Convolution with 3*3 horizontal line segment mask
(linear output)



PIC37 Convolution with 3*3 high pass filter (linear output)

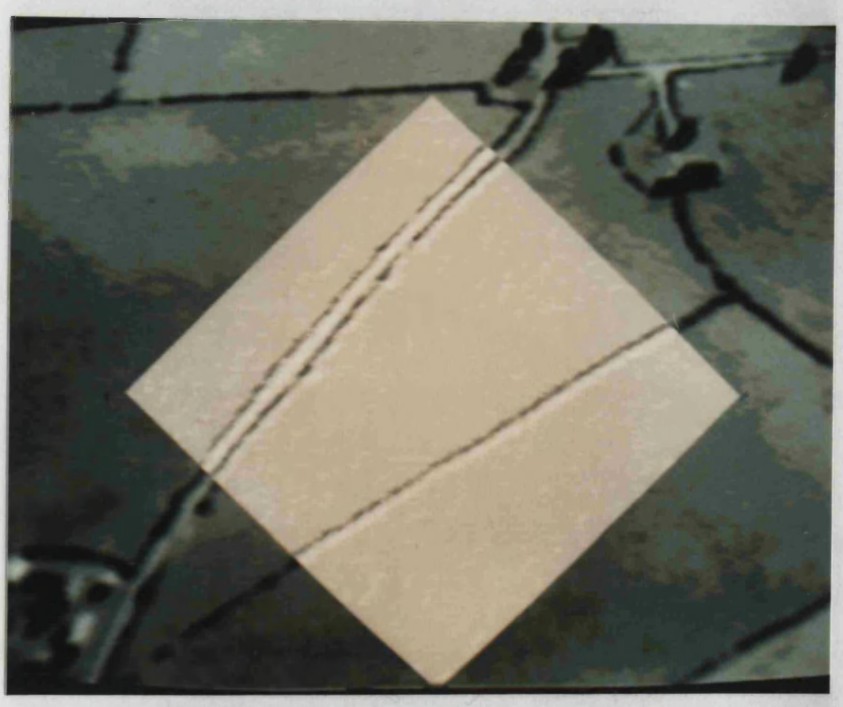
PIC38 Convolution with 3*3 low pass filter, shown after 4 recursions (linear output)



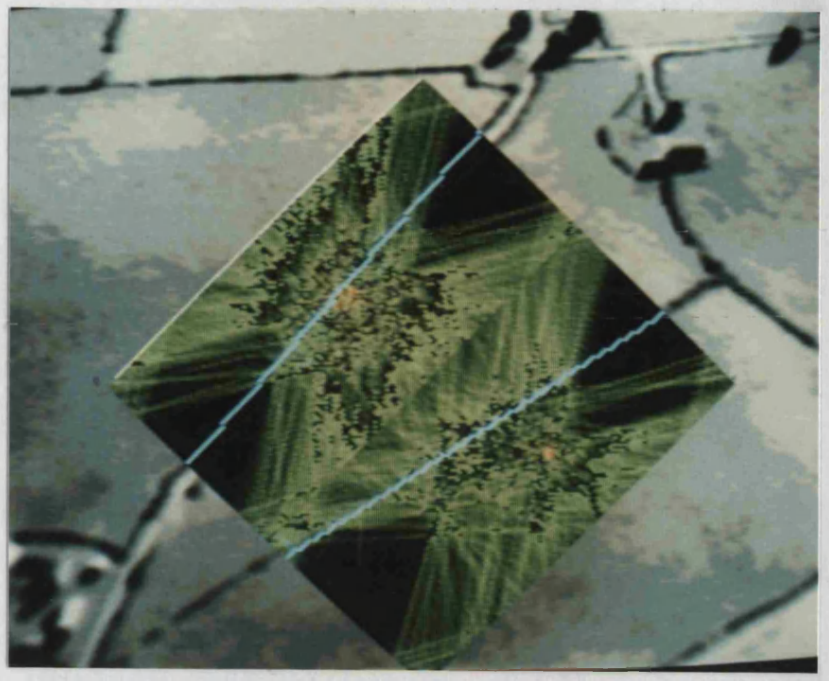
PIC39 Median filtering, shown after 4 recursions
(linear output)

PIC40 Maximum filtering, shown after 4 recursions
(linear output)

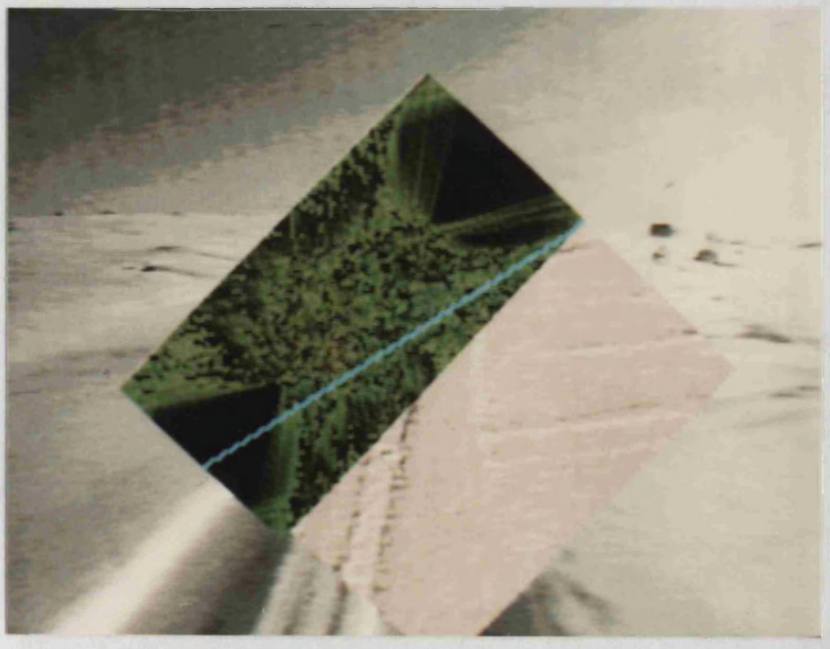
PIC41 Minimum filtering, shown after 4 recursions (linear output)



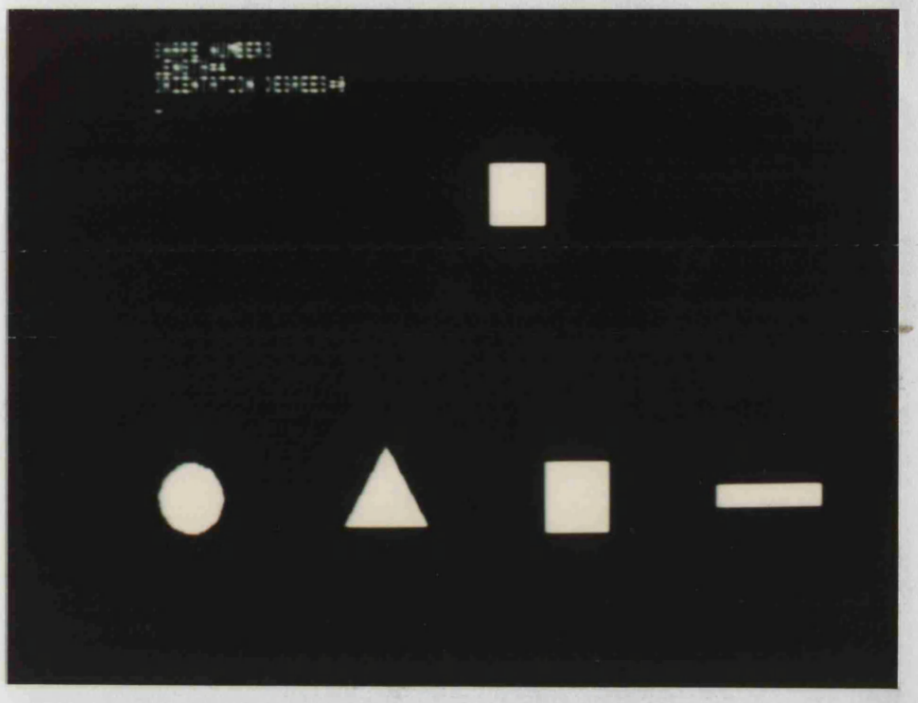
PIC42 Vector Scanning -45° edge enhancement of roads on a terrain model (linear output)



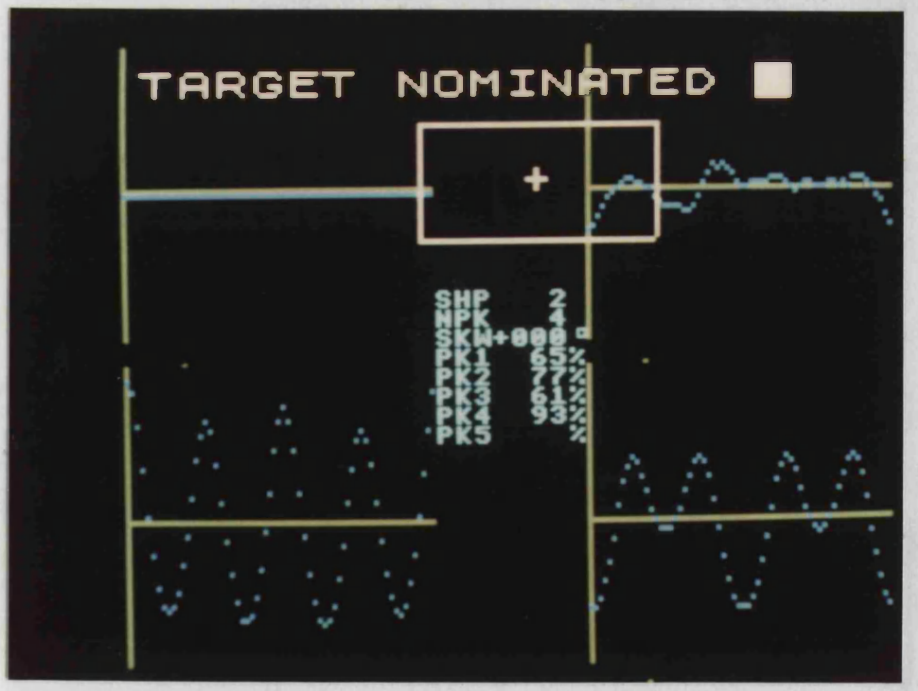
PIC43 Roads found on a terrain model using Hough transforms. Blue lines are computed line equations, green boxes are overlaid accumulator arrays.



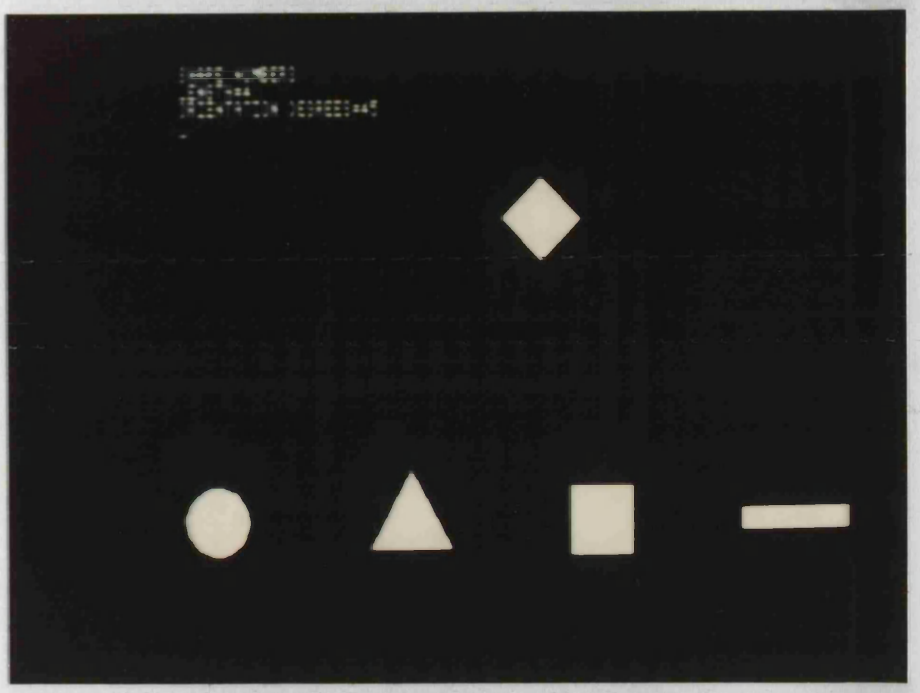
PIC44 Mean direction of snow tracks found using Hough transforms.



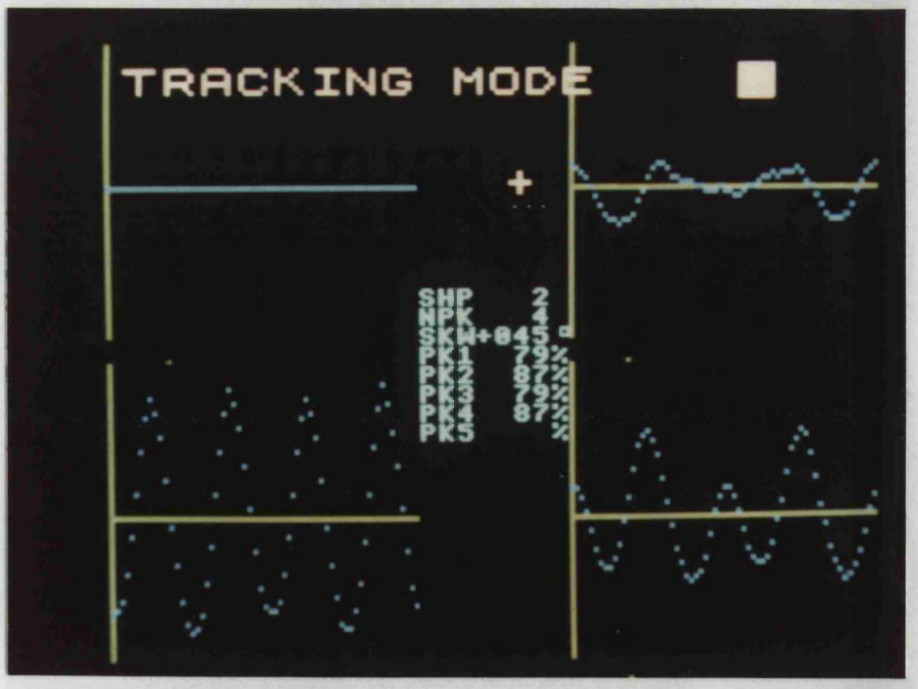
PIC45 Unskewed square target shown with model set.



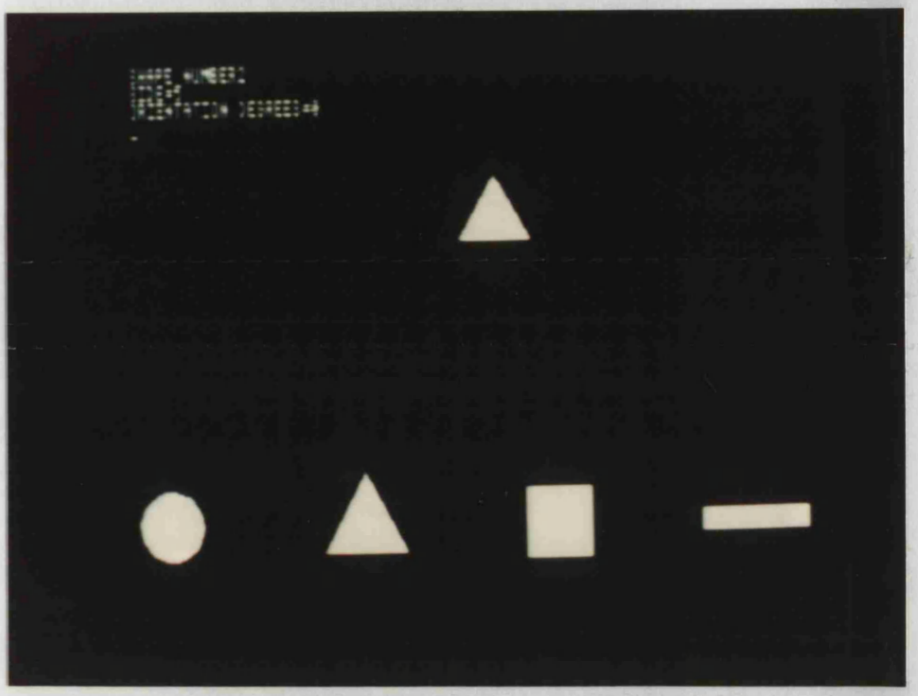
PIC46 Results from the vector scanning shape descriptor for the unskewed square.



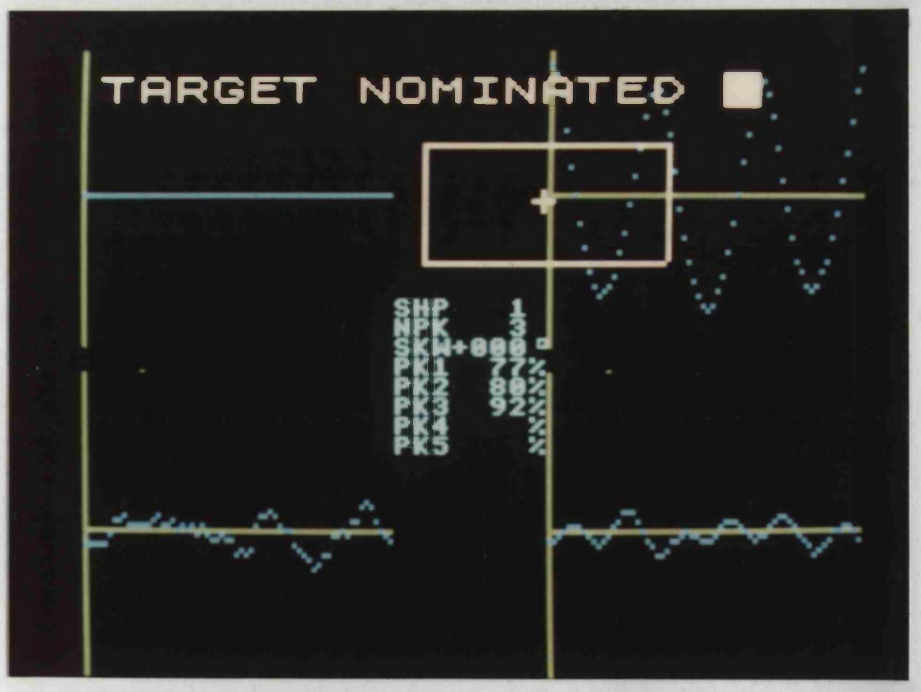
PIC47 Square target with 45° skew shown with model set.



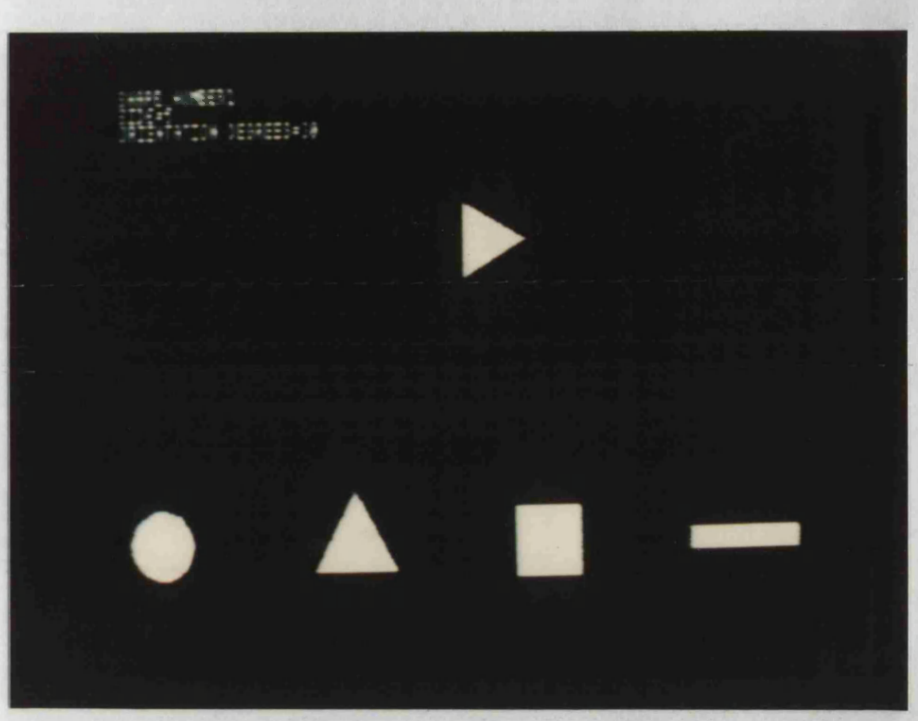
PIC48 Results from the vector scanning shape descriptor for the 45° skewed square.



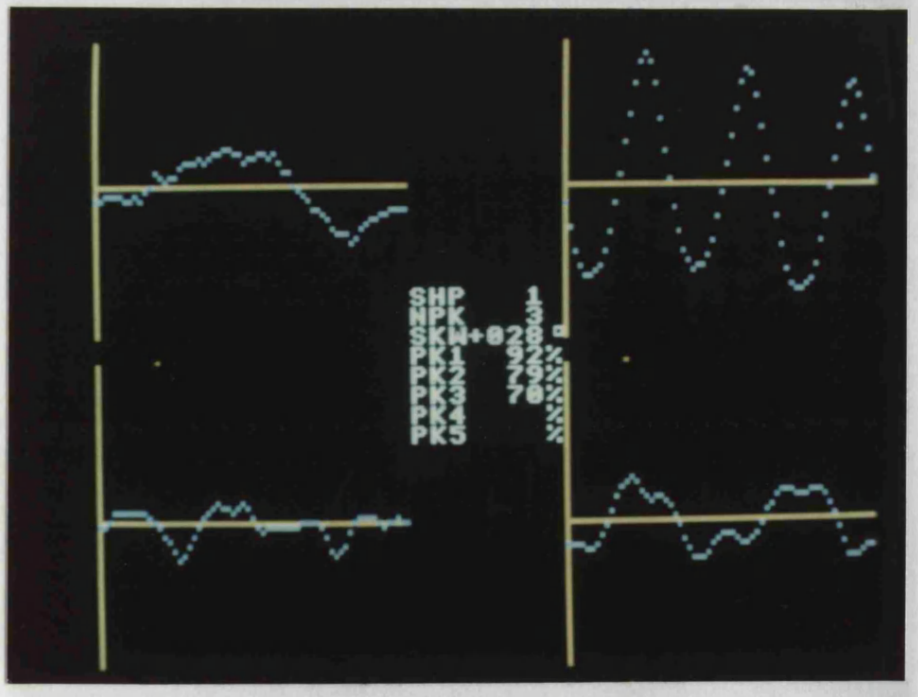
PIC49 Unskewed triangular target shown with model set.



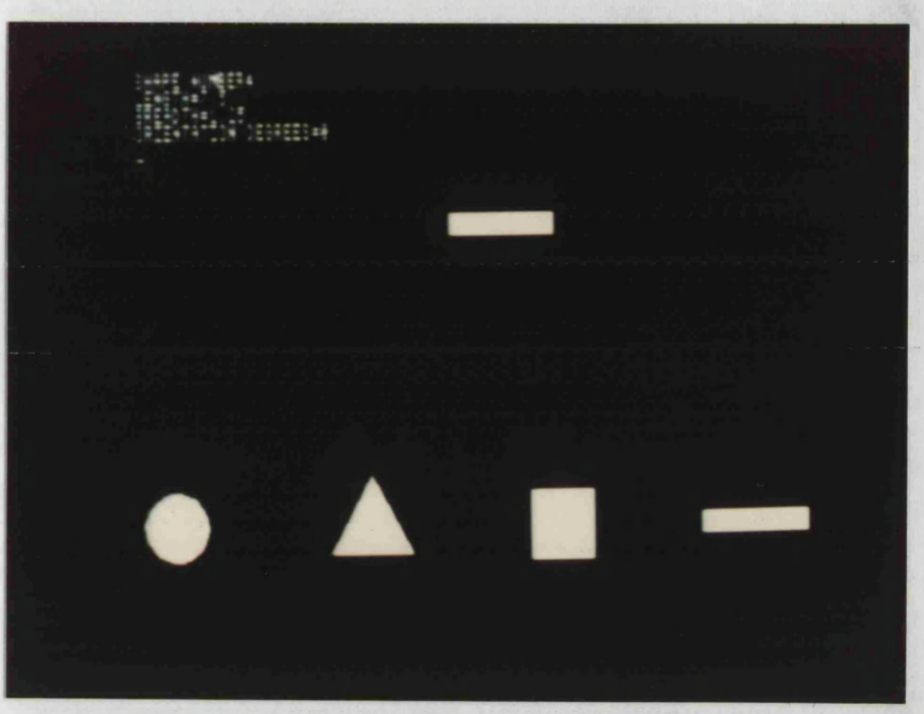
PIC50 Results from the vector scanning shape descriptor for the unskewed triangle.



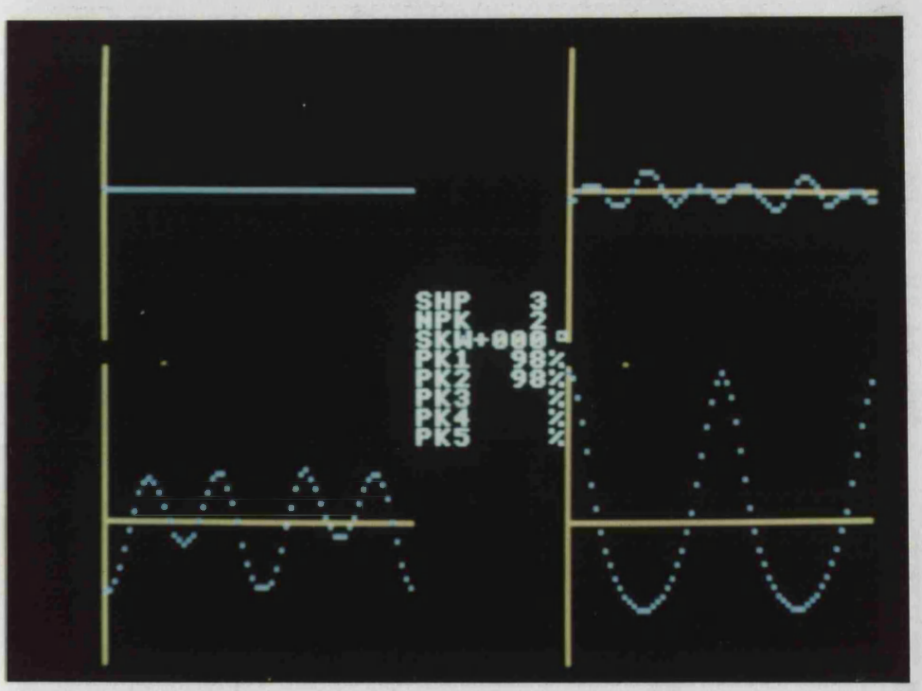
PIC51 Triangle skewed by 30° shown with model set.



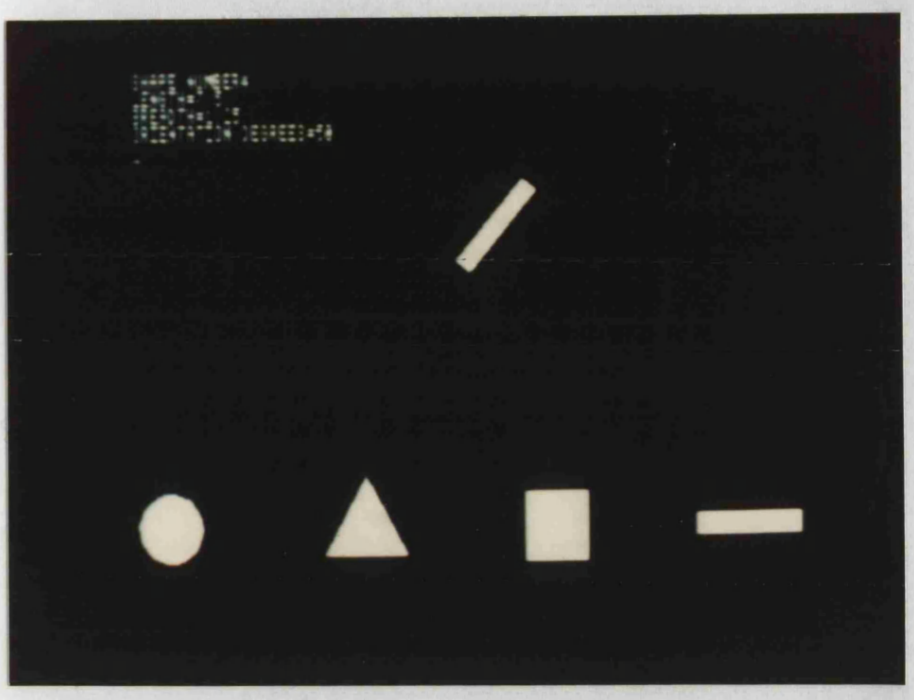
PIC52 Results from the vector scanning shape descriptor for the 30° skewed triangle.



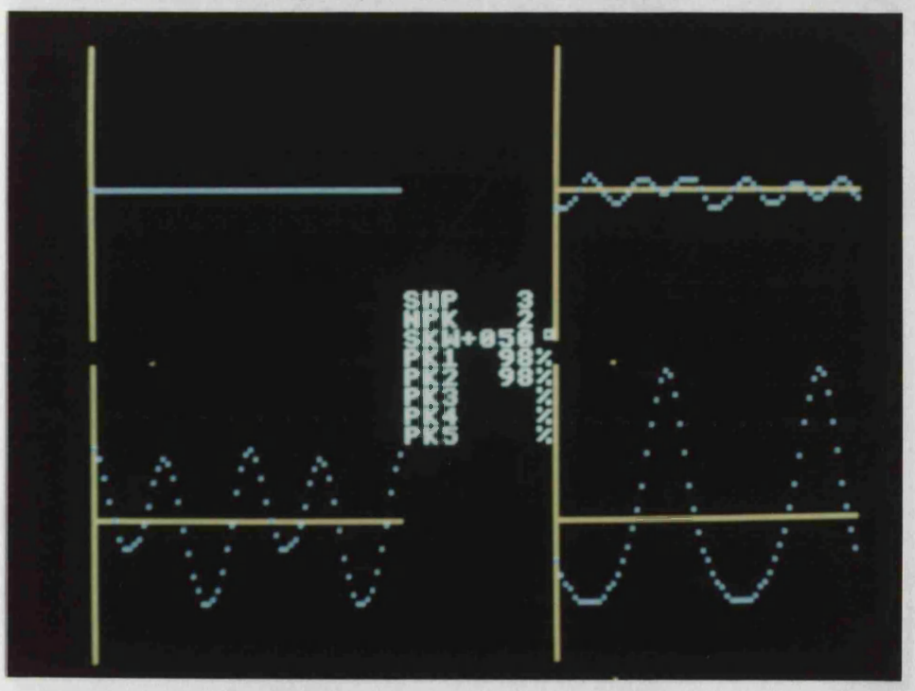
PIC53 Unskewed bar target shown with model set.



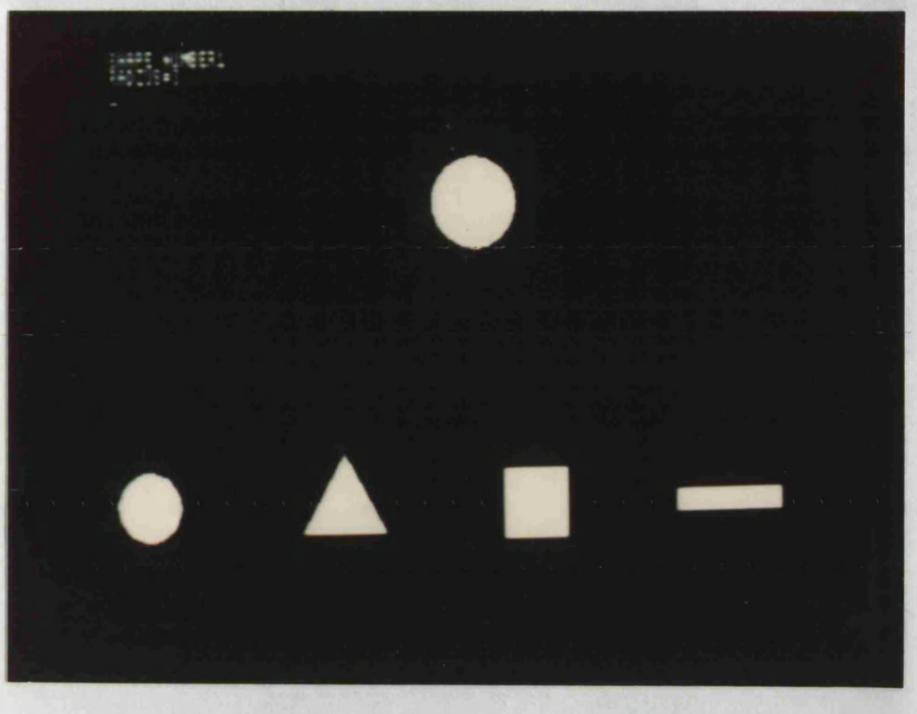
PIC54 Results from the vector scanning shape descriptor for the unskewed bar.



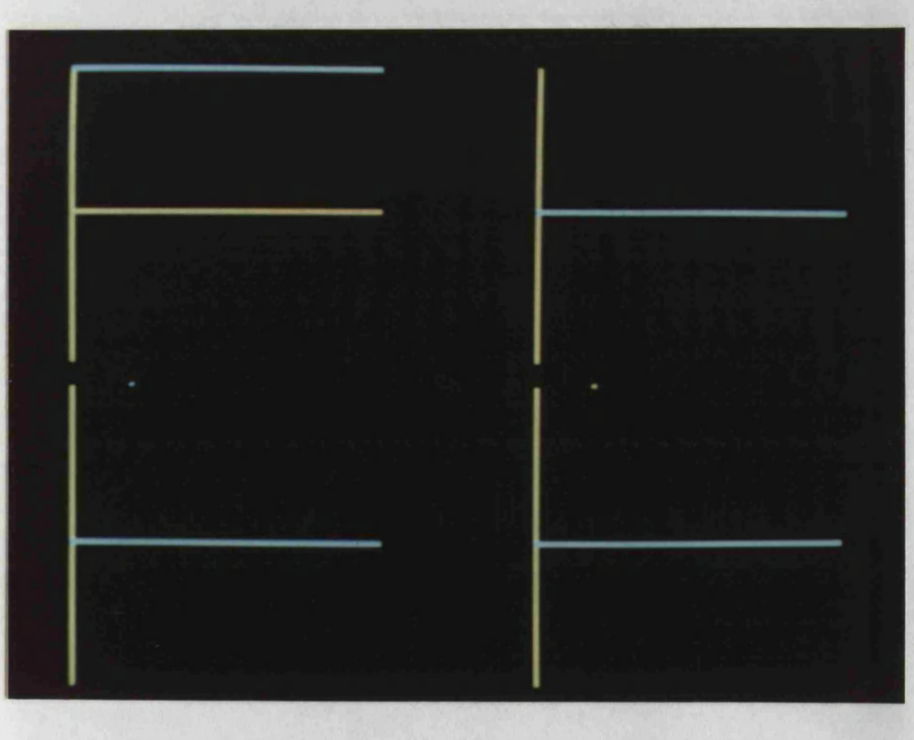
PIC55 Bar skewed by 50° shown with model set.



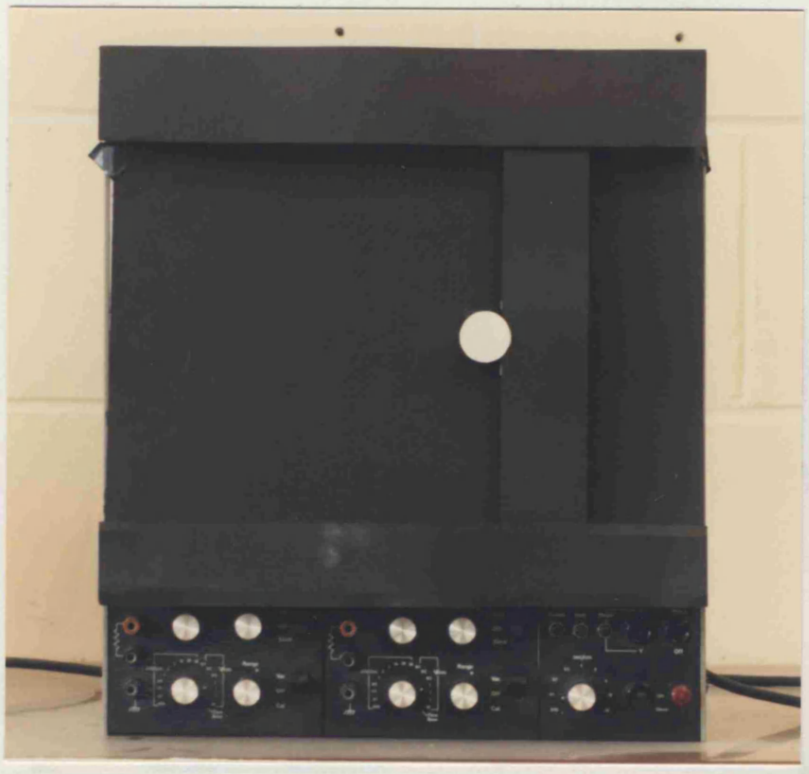
PIC56 Results from the vector scanning shape descriptor for the 50° skewed bar.



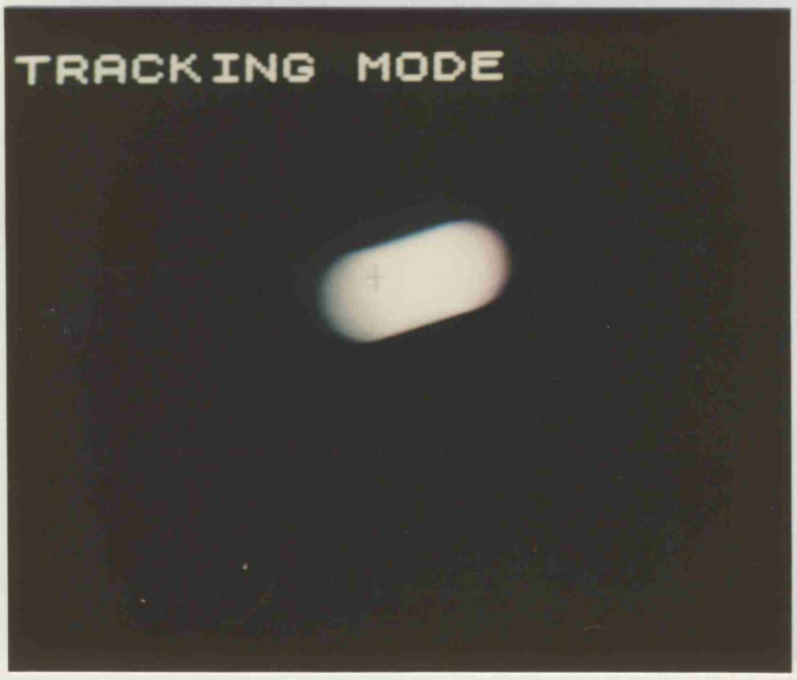
PIC57 Circular target shown with model set.



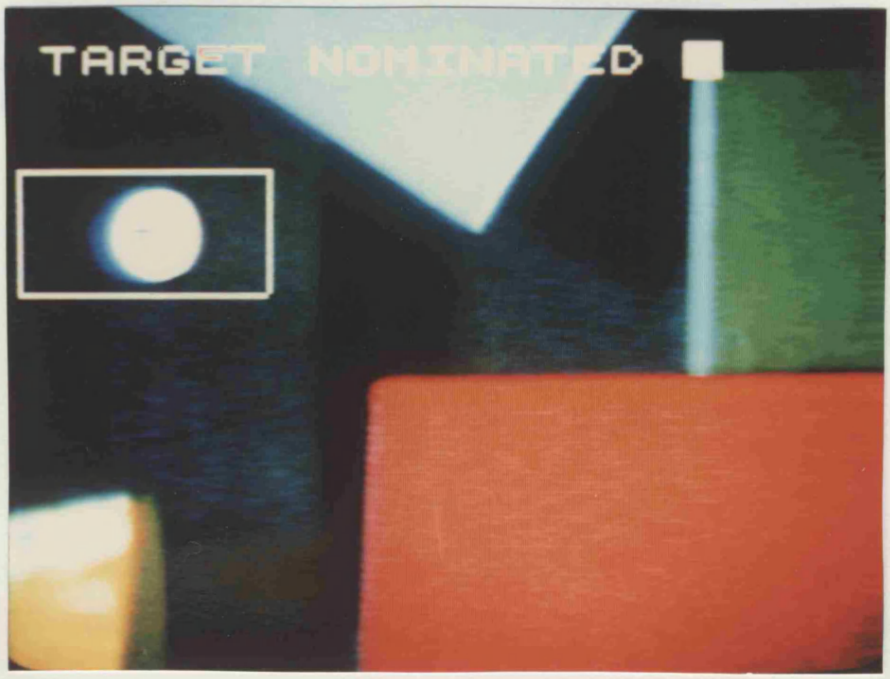
PIC58 Results from the vector scanning shape descriptor for the circle.



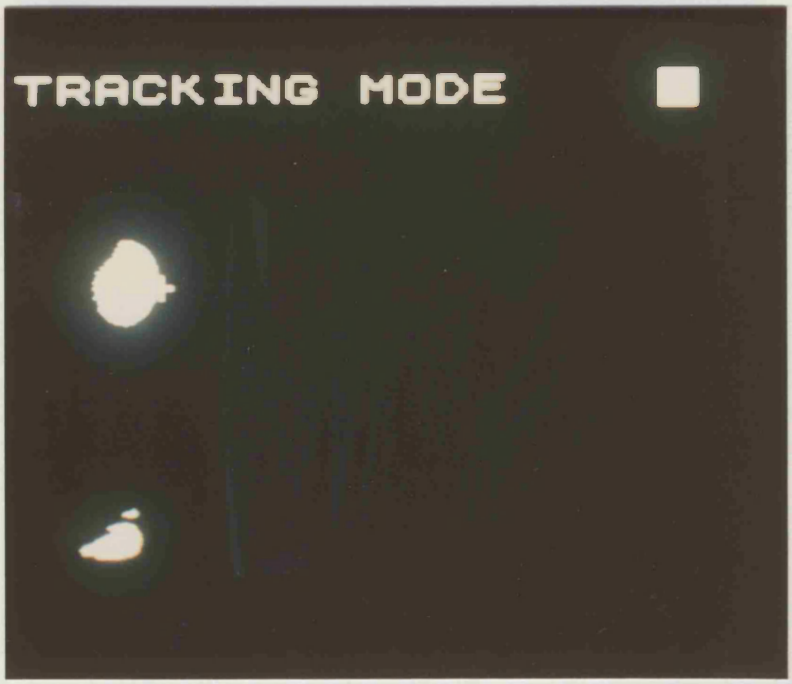
PIC59 Single physical target test controller.



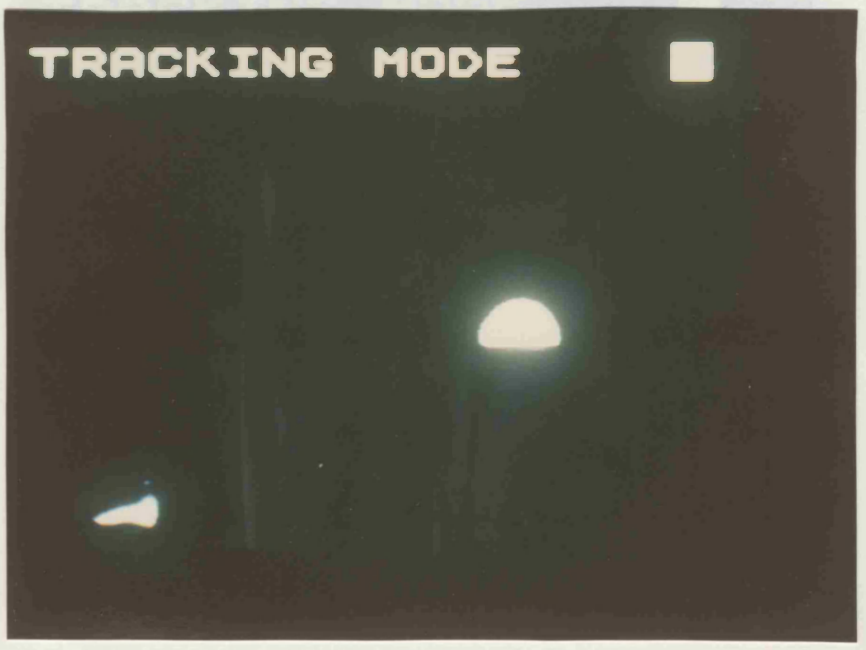
PIC60 Single physical target being tracked at frame rate by VOSTTAC1 (time lapsed photograph).



PIC61 Moving target within cluttered environment.

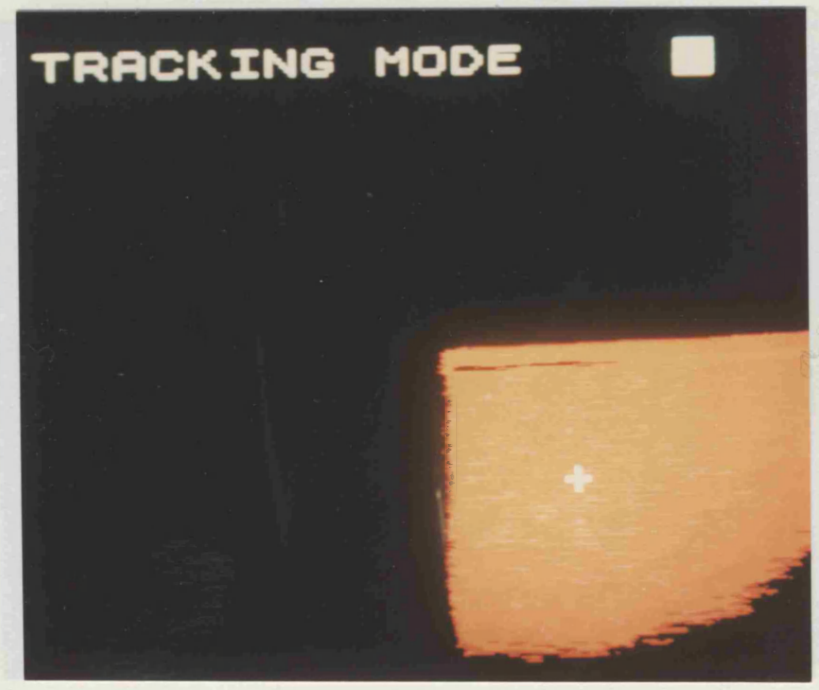


PIC62 Result of multidimensional colour thresholding on target in PIC61.



PIC63 Target of PIC61 having dipped behind book in bottom right hand corner.

PIC64 Folder at the top of PIC61 nominated instead of target.

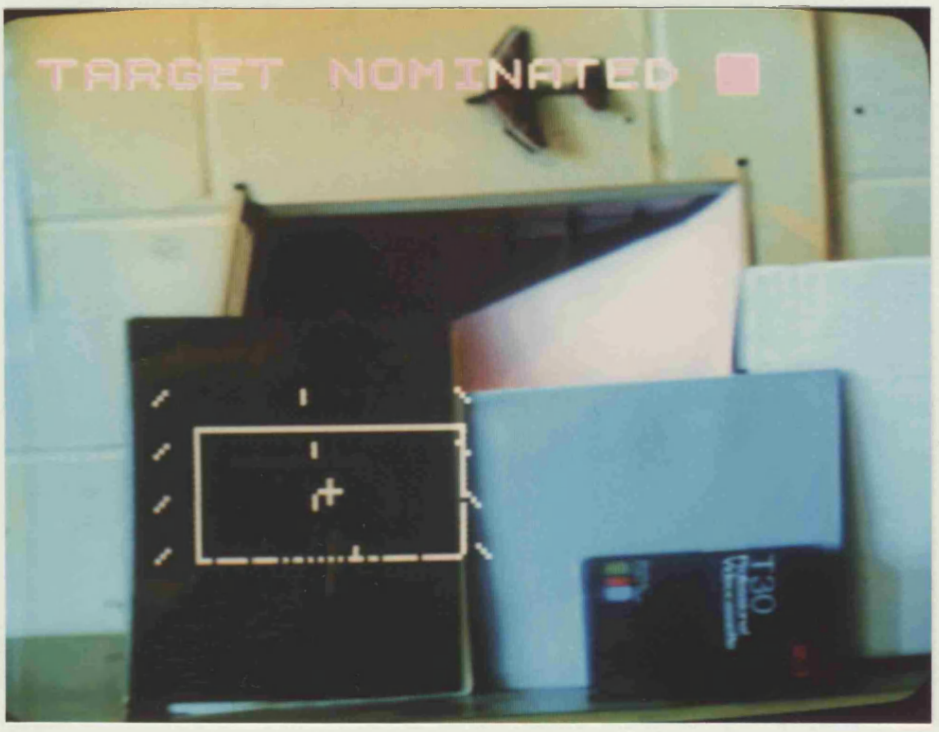


PIC65 Book in PIC61 nominated instead of target.

PIC66 Colour test image.

PIC67 Thresholded blue file and a candid view of a careless photographer!

PIC68 Pathfinder technique used to find the central, right and left lines of the nominated area.



PIC69 Pathfinder technique used to outline extremities and centre of the black file.

PIC70 Road section taken from video taped recording.

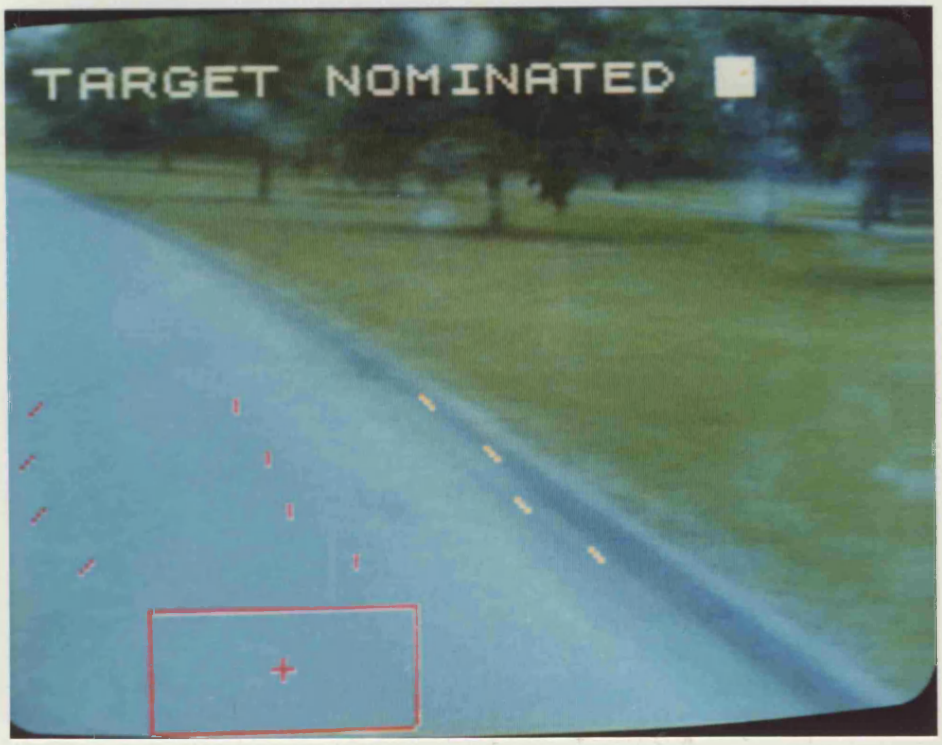
PIC71 Road scene shown after single peak multidimensional thresholding.

PIC72 Road scene shown after double peak multidimensional thresholding.

PIC73 Pathfinder generating "go left" vectors on approach to a right hand road edge.

PIC74 Pathfinder fairly contented !i.e. going straight.

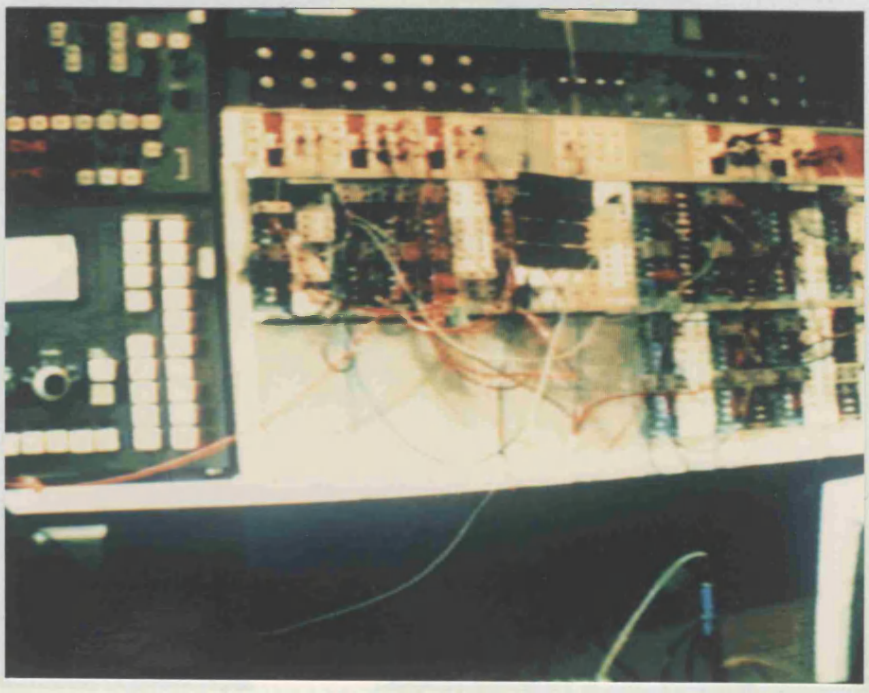
PIC75 Pathfinder generating "go right" vectors on approach to a left hand curb.



PIC76 Pathfinder requiring a slight leftwards course correction.

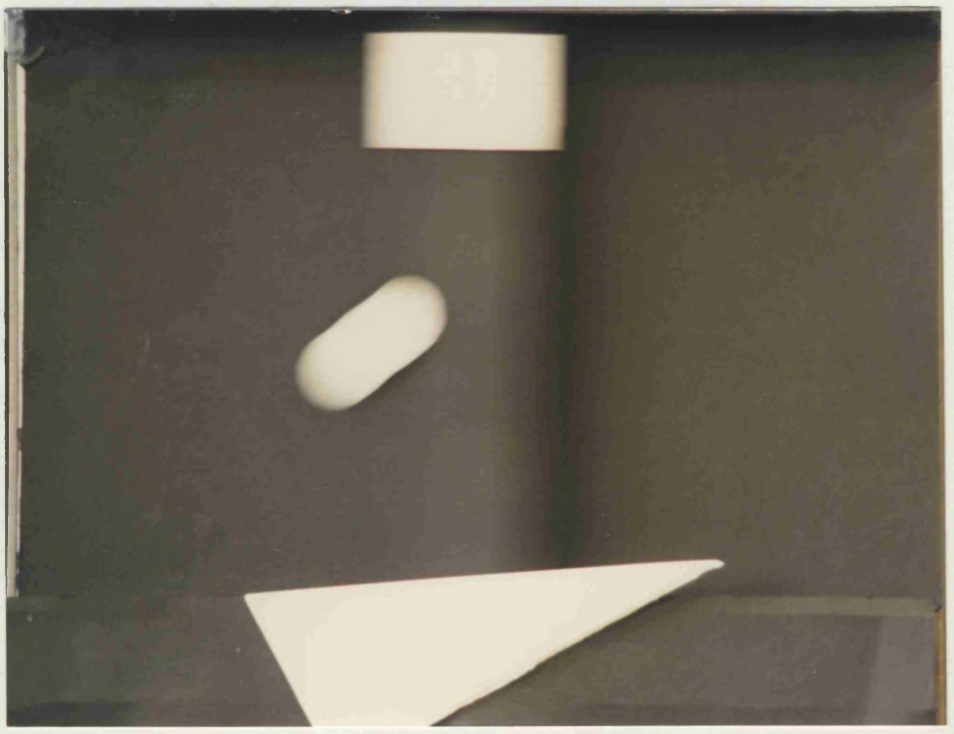
PIC77 A non-descript side of a room !

PIC78 Person (with a pipe !) shown movement detected against the background of PIC77.

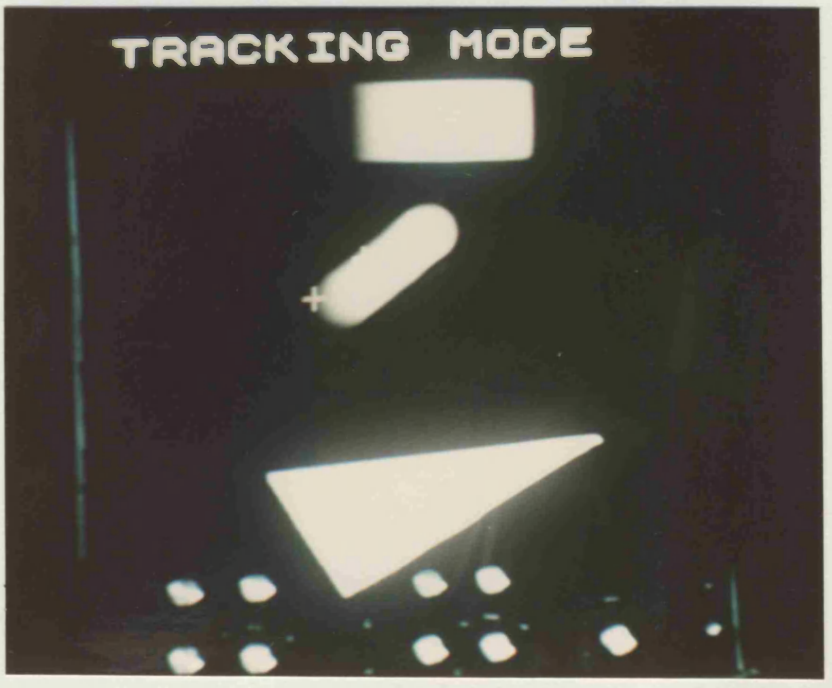


PIC79 A nostalgic view of an analogue computer with a barely perceptible swinging pendulum at the top.

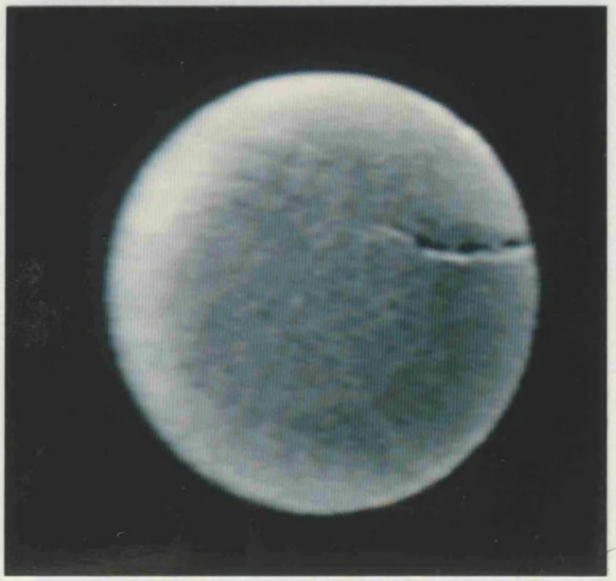
PIC80 The pendulum of PIC79 revealed by movement detection.



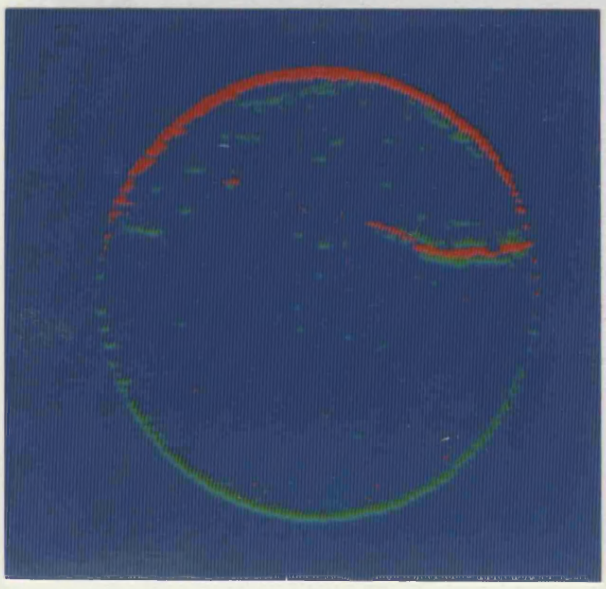
PIC81 Physical production of a multitarget scene.



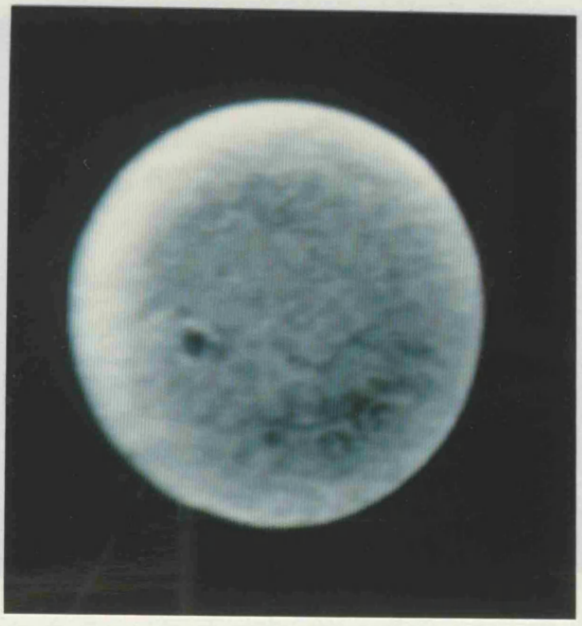
PIC82 Tracking of a specified target in the presence of others.



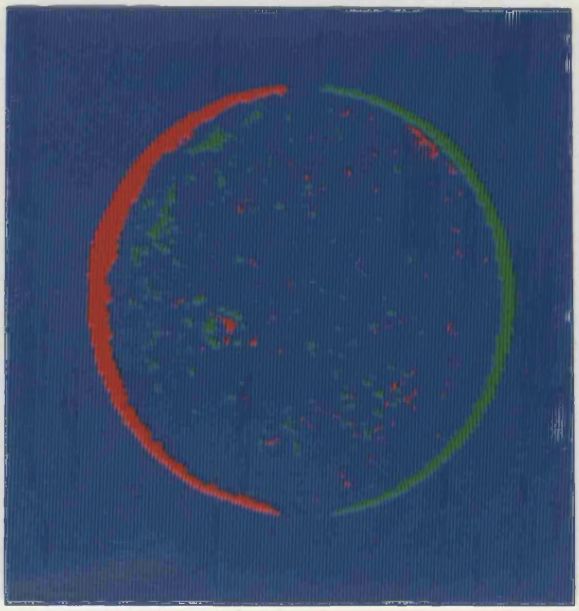
PIC83 Aluminium slug with a side split.



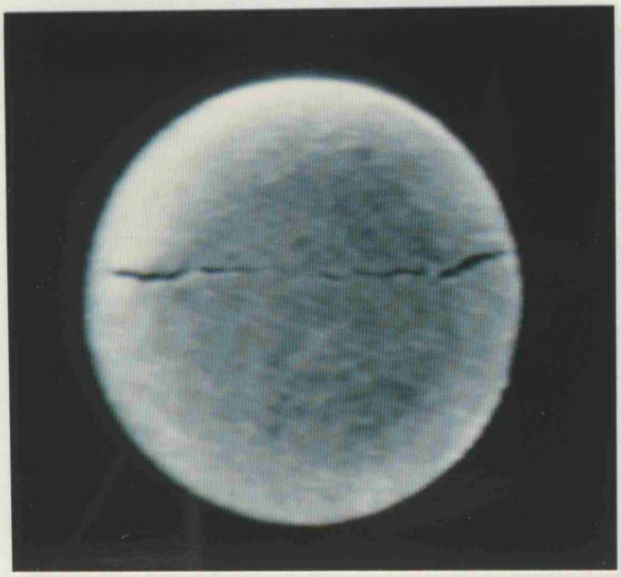
PIC84 Enhanced side split.



PIC85 Aluminium slug with indentations/pips.

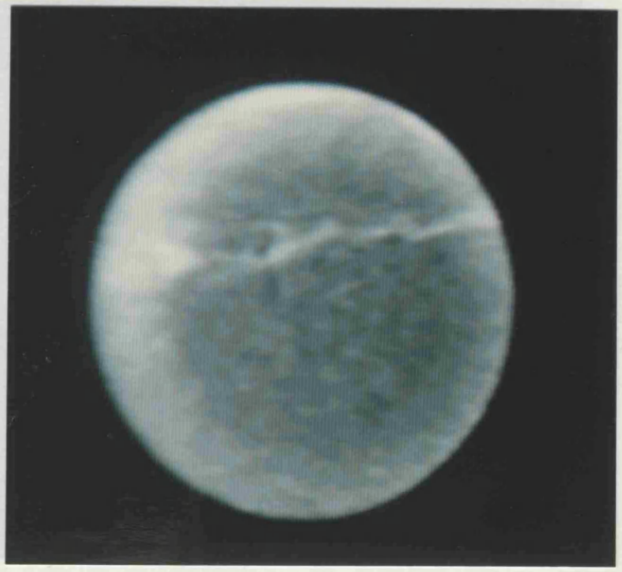


PIC86 Enhanced indentations/pips.

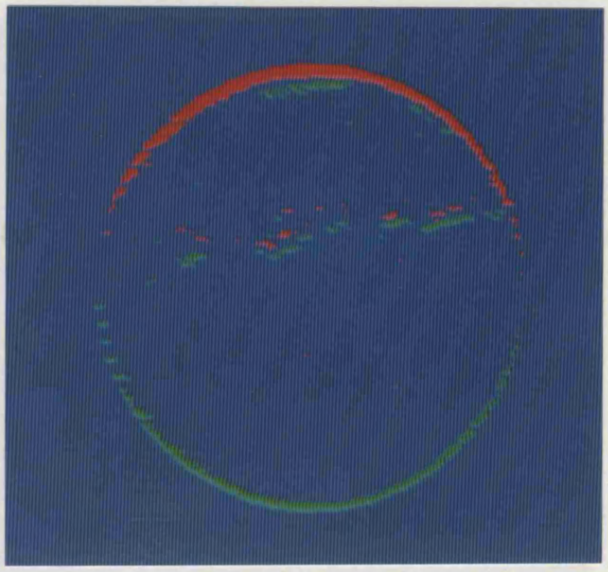


PIC87 Aluminium slug with line crack.

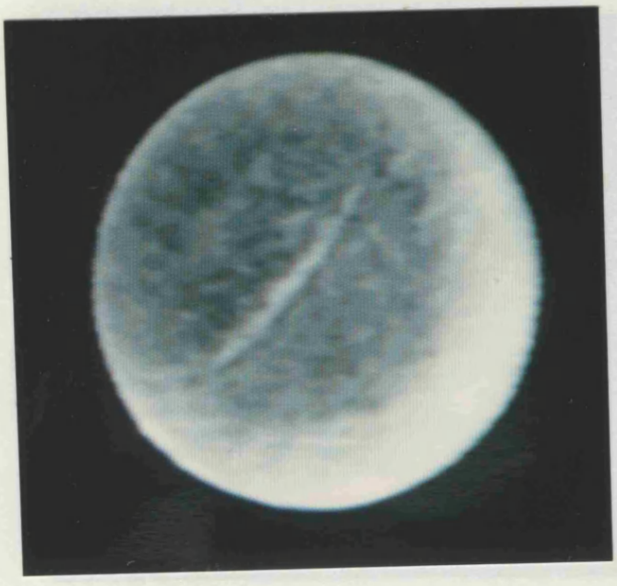
PIC88 Enhanced line crack.



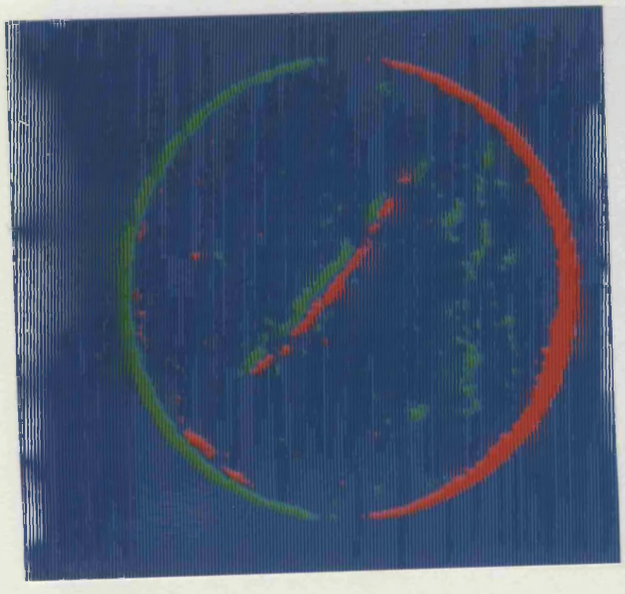
PIC89 Aluminium slug with fault line.



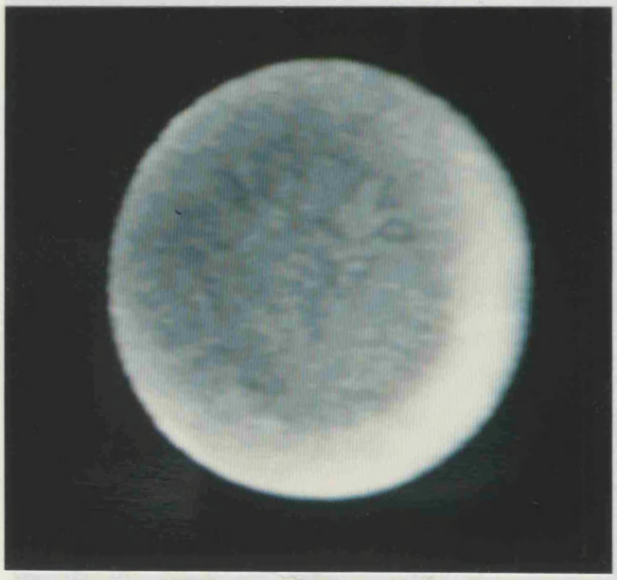
PIC90 Enhanced fault line.



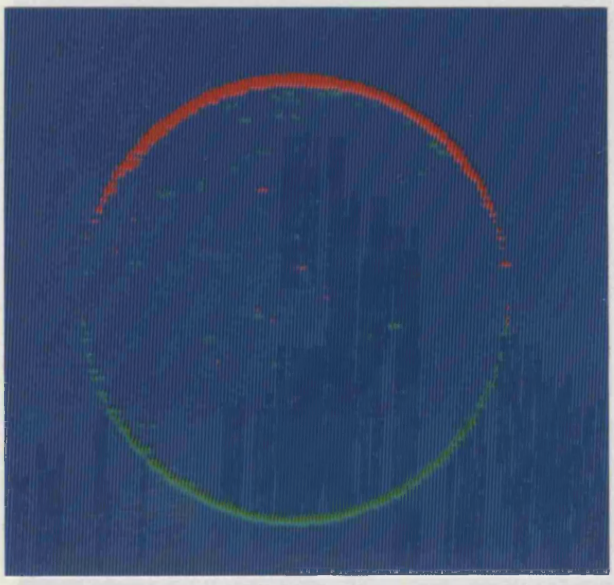
PIC91 Aluminium slug with a cut.



PIC92 Enhanced cut.

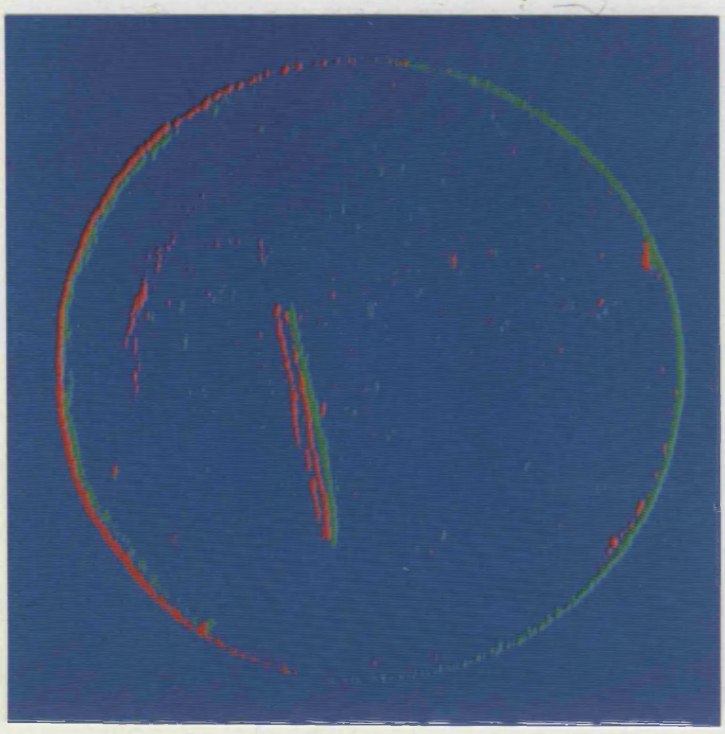


PIC93 Defect free aluminium slug.

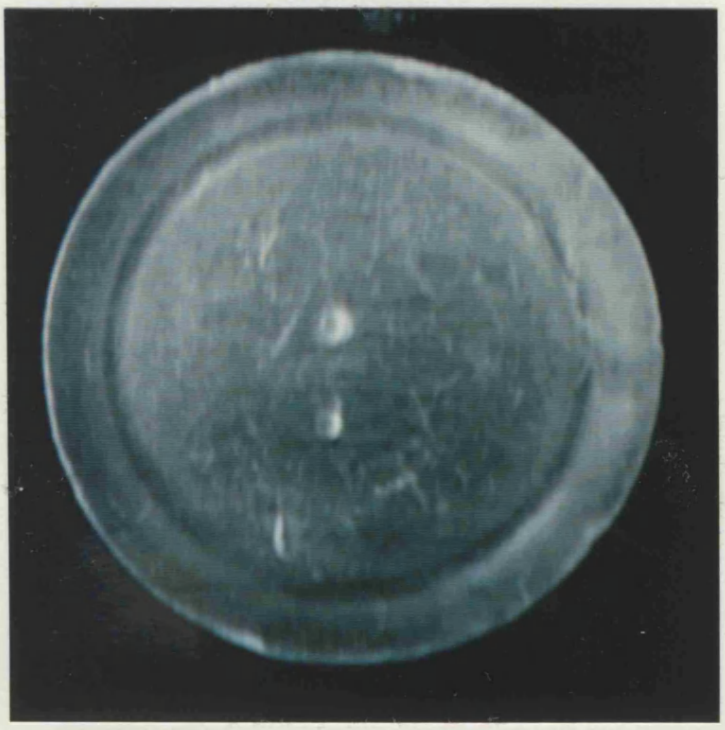


PIC94 Slug after enhancement.

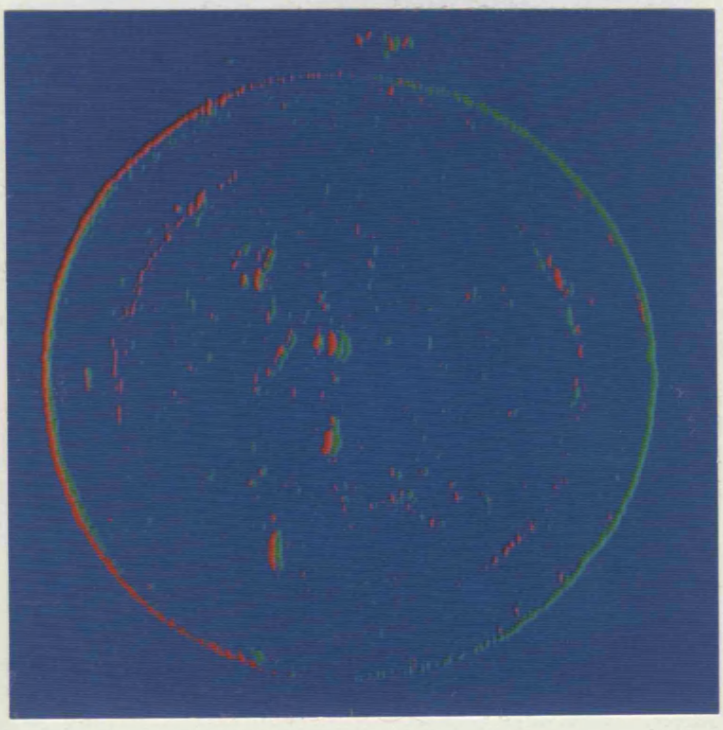
PIC95 Large aluminium disc with deep scratch.



PIC96 Enhanced scratch.



PIC97 Large aluminium disc with surface indentations.



PIC98 Enhanced indentations.